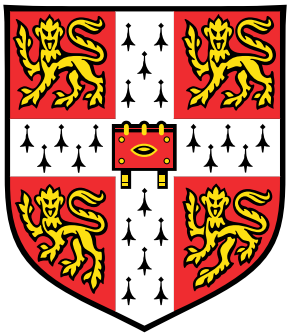
Optimal anticipatory control as a theory of motor preparation



Ta-Chu Kao

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Pembroke College

March 2021

Contents

To my loving parents

Declaration

I hereby declare that this thesis is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the preface and specified in the text. It is not substantially the same as any work that has already been submitted before for any degree or other qualification except as declared in the preface and specified in the text. I further state that no substantial part of my thesis has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the preface and specified in the text. This thesis contains 39,543 words including appendicies, bibliography, footnotes, tables, and equations and has 27 figures and 3 tables. Partial accounts of this thesis have been published in the following collaborative papers:

Ta-Chu Kao and Guillaume Hennequin (2019). Neuroscience out of control: control-theoretic perspectives on neural circuit dynamics. *Current Opinion in Neurobiology*, 58, 122–129.

Ta-Chu Kao, Mahdieh S. Sadabadi and Guillaume Hennequin (2021). Optimal anticipatory control as a theory of motor preparation: a thalamo-cortical circuit model. *Neuron* (in press).

Kristopher T. Jensen, **Ta-Chu Kao**, Marco Tripodi, and Guillaume Hennequin (2020). Manifold GPLVMs for discovering non-Euclidean latent structure in neural data. *Advances in Neural Information Processing Systems*, 33, 22580–22592.

Acknowledgements

This thesis would not have been possible but for the incredible support that I have had the good fortune of receiving over the last four years.

First, I would like to extend my deepest gratitude to my supervisor Guillaume, who has taught me more things than I can remember. His tireless guidance and support has helped me stay the course, and I could not have hoped for a better supervisor.

I am also extremely thankful for my lab mates, in particular Kris and Marine, who I have collaborated with on multiple projects. Their comments on this thesis have guided me in my countless revisions and are most appreciated.

It would be remiss of me not to extend my sincere thanks to the Biological Learning Group (BLG). Our bi-weekly lab meetings and journal clubs have been a constant source of inspiration. I am especially grateful to my advisor Máté and fellow BLG members Jake, Alberto, and Rodrigo for their insightful comments and suggestions on my work over the last four years.

Special thanks to fellow CBL members Robert, Adrià, Jonathan, and Will for fascinating discussions about a variety of topics in machine learning and science in general over the course of my PhD.

I am also grateful to Karel Svoboda for hosting me at Janelia November 2020, which was an incredible learning experience. Many thanks to Arseny and Lorenzo not only for their comments on my work but also for the fruitful discussions we had about motor control when I was at Janelia.

Thanks also to the students and faculty members at the Cajal computational neuroscience summer school, with whom I had many thought-provoking scientific discussions.

Last but by no means least, without my parents' unwavering support and Caroline's constant good humour, I would not have been able to complete this thesis. I simply cannot overstate the importance of their patience and encouragement.

Optimal anticipatory control as a theory of motor preparation

Ta-Chu Kao

Supported by a decade of primate electrophysiological experiments, the prevailing theory of neural motor control holds that movement generation is accomplished by a preparatory process that progressively steers the state of the motor cortex into a movement-specific optimal subspace prior to movement onset. The state of the cortex then evolves from these optimal subspaces, producing patterns of neural activity that serve as control inputs to the musculature. This theory, however, does not address the following questions: what characterizes the optimal subspace and what are the neural mechanisms that underlie the preparatory process? We address these questions with a circuit model of movement preparation and control. Specifically, we propose that preparation can be achieved by *optimal feedback control* (OFC) of the cortical state via a thalamo-cortical loop. Under OFC, the state of the cortex is selectively controlled along state-space directions that have future motor consequences, and not in other inconsequential ones. We show that OFC enables fast movement preparation and explains the observed orthogonality between preparatory and movement-related monkey motor cortex activity. This illustrates the importance of constraining new theories of neural function with experimental data. However, as recording technologies continue to improve, a key challenge is to extract meaningful insights from increasingly large-scale neural recordings. Latent variable models (LVMs) are powerful tools for addressing this challenge due to their ability to identify the low-dimensional latent variables that best explain these large data sets. One shortcoming of most LVMs, however, is that they assume a Euclidean latent space, while many kinematic variables, such as head rotations and the configuration of an arm, are naturally described by variables that live on non-Euclidean latent spaces (e.g., SO(3) and tori). To address this shortcoming, we propose the Manifold Gaussian Process Latent Variable Model, a method for simultaneously inferring nonparametric tuning curves and latent variables on non-Euclidean latent spaces. We show that our method is able to correctly infer the latent ring topology of the fly and mouse head direction circuits.

Chapter 1

Introduction

Advances in systems neuroscience over the last twenty years bring with them new questions and fresh challenges in the study of motor control. Chief among these is a need for a theory of biological motor control that can explain large-scale behavioural and neuronal recordings of animals performing complex motor tasks. A second but equally important challenge is a need for technical tools to process and analyse these experimental recordings. This need is especially pressing as the data sets increase in size and become more heterogeneous as recording technologies improve. This thesis includes contributions in both of these directions. In particular, we propose a model of movement preparation and generation that explains key aspects of monkey motor cortex population activity during a delayed reach task in Chapter 3. In Chapter 5, we extend Gaussian Process Latent Variable Models to discover non-Euclidean latent structures in neuronal recordings. Here, I review these two contributions in the context of relevant literature.

1.1 Towards a theory of biological motor control

The ability to accurately plan and robustly produce movements is one of the most important computations that the brain performs. Without it, animals are unable to eat, communicate, reproduce, change their environments, or escape from danger. Ironically, the importance of motor control is easily taken for granted, obfuscated by the relative ease with which we perform daily movements, and yet most acutely felt when that ability is suddenly lost.

The importance of motor control notwithstanding, its neural mechanisms remain a mystery due largely to its complexity. Several factors contribute to the complexity of motor control. First, motor control encompasses multiple sub-computations including muscular control [84], motor planning, motor prediction, state estimation [205, 206], and motor learning. Each sub-

computation may employ a different algorithm or neural implementation depending on the type of movement produced: voluntary or involuntary, rhythmic or ballistic [84].

Second, motor computations are distributed and hierarchical, involving multiple brain areas, ranging from low-level control of the musculature in the spinal cord and the midbrain [126, 153] to the high-level planning in the cortex [54, 166], thalamus, basal ganglia [135], and the cerebellum [53, 208]. A complete theory of motor control thus necessarily involves understanding the interaction between multiple brain regions, the animal's biomechanics and the environment.

Third, the motor system is degenerate on multiple levels. At the level of motor planning, the same goal (e.g., picking up a cup of coffee) can be achieved by an infinite number of different movements (e.g., hand trajectories through space). At the level of muscular control, the same patterns of muscle activity can be achieved by many different patterns of neural activity in the brain. Characterising such degeneracies is challenging, as we cannot, with today's technology, record all the neurons that may be involved in a single motor computation.

Last but not least, motor computations are dynamic in nature, requiring an understanding of how populations of neurons collectively perform computations over time. This adds an additional layer of complexity when compared to the study of static sensory representations, which can often be studied without taking recurrence and feedback into consideration.

While the challenges involved in understanding motor control are great — and at times, may even seem insurmountable — new advances in recording technologies, coupled with the advent of machine learning and improved computational capacity gives us hope that we can one day understand the neural mechanisms of motor control. In fact, there are many reasons why this is the *best of times* to be studying motor control. To start with, we are now simultaneously recording more neurons, across more brain regions, at higher temporal resolutions, and with animals performing more complicated motor tasks than ever before [202]. This has produced high quality neural recordings, which provide us with unprecedented access to the inner workings of the brain areas that underlie motor control. What is more, due in part to Moore's law and the development of specialised computer chips such as graphics processing units, we now have the computational capacity to run increasingly complicated analyses on large-scale neural and behavioural recordings [35, 131]. Correspondingly, the success of machine learning in solving difficult — some seemingly impossible — problems in a plethora of domains has brought exciting new methods and ideas to bear on the study of motor control.

Therefore, an important motivation for this thesis is to build on these exciting new advances in systems neuroscience and contribute towards a theory of biological motor control that is

grounded in experimental studies.

1.1.1 Computational building blocks

A first step towards a theory of biological motor control is to elucidate the relevant computations that the brain needs to perform to effectively control movements. Five of the most important computations are [205, 206]:

- *motor planning*: planning an upcoming movement to achieve a specific goal (e.g., pick up an object),
- *inverse control*: determining the sequence of motor commands (e.g., muscle contractions) necessary for executing that plan,
- *motor prediction*: predicting the effect that a sequence of motor commands has on the effector (e.g., the position of the arm after applying a sequence of muscular contractions; this may be achieved for example with a forward model of the body biomechanics),
- *state estimation*: estimating the state of the body (e.g., position of the hand) by combining sensory information and predictions of a forward model, and
- *motor learning*: learning new motor skills and improving old ones through experience.

Over the last twenty years, clever behavioural experiments have provided compelling evidence that the brain performs all of the aforementioned computations and in many situations performs them *optimally*. That is, these studies have shown that animal behaviours are consistent with those predicted by optimal control theory [159, 185, 187] and Bayesian probability theory (e.g., animals perform Bayesian inference to estimate the state of the body; 93, 116). Notably, Todorov and Jordan applied stochastic optimal feedback control to model motor planning in the face of uncertainty. They showed through simulations that the optimal strategy is to correct movement variability in dimensions that matter for the task, and allow behavioural variability to persist or even grow in task-irrelevant dimensions [187]. They found that human behaviour is consistent with this prediction when performing a range of behavioural tasks.

1.1.2 Motor cortex controls muscles

Despite the success of behavioural experiments at elucidating the principles of neural motor computations, they do not explain the neural mechanisms by which these computations are

carried out inside the brain. This raises several questions. Where do the motor commands to the musculature arise in the brain and how are they generated?

Electrophysiological studies suggest that the motor cortex generates at least some of the motor commands that actuate the musculature. As early as 1870, Fritsch and Hitzig demonstrated that electrical stimulation of the dog motor cortex elicited movements [62]. Later, Evarts and colleagues showed that the neural activity in the motor cortex are correlated with force production [39, 40, 180]. Moreover, the existence of the corticospinal pathway (i.e., motor cortex neurons that directly project to motor neurons and interneurons in the spinal cord) suggests that the motor cortex could directly exert muscular control if required [61, 127]. These studies suggest that the primary role of the motor cortex is to actuate the musculature.

Georgopolous and colleagues proposed an alternative view, namely that the motor cortex represents high-level motor plans, which are converted into commands in the spinal cord. Indeed, neurons in the motor cortex of reaching monkeys are selective to high-level movement parameters such as reach direction [55] and movement speed [120]. Can the two seemingly contradictory views be reconciled? Todorov and later Lillicrap and Scott showed through computational modelling that the findings of Georgopolous and colleagues' are expected, if one takes into account the biomechanics of the effector [100, 183]. In other words, the tuning of cortical neurons to high-level movement parameters is an epiphenomenon that results from the motor cortex controlling a limb with its own mechanics.

Ultimately, whether the motor cortex is directly involved in muscular control may depend on the movement type and the context in which the movement is produced. For example, Kawai et al. showed that mice need their motor cortices to learn a level-pressing task, but not for executing it [89]. This study suggests that the role of the motor cortex may extend beyond simple muscular control, and in some contexts, other brain regions can control motor outputs independently of the motor cortex.

1.1.3 Motor cortex as a dynamical system

A potential mechanism by which the motor cortex generates motor commands is through its population dynamics, an idea commonly referred to in the literature as the *dynamical systems perspective* of the motor cortex [21, 166]. Concretely, let the activity of a population of N neurons in the motor cortex be represented by a vector $\boldsymbol{x}(t) = (x_1(t), x_2(t), \ldots, x_N(t))$ where $x_i(t)$ is the activity of neuron i at time t. The activity of the population evolves according to

$$\frac{d\boldsymbol{x}}{dt} = f(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{1.1}$$

where $\boldsymbol{u}$ denotes inputs to the population, and f is some function of $\boldsymbol{x}$ and $\boldsymbol{u}$. The activity of this population may then be converted into motor commands $\boldsymbol{m}(t) = g(\boldsymbol{x}(t))$ through some mapping g that models the effect of downstream spinal cord circuitry.

The dynamical systems perspective of the motor cortex is typically contrasted with a representational view of motor cortex activity, proposed by Georgopoulos and colleagues:

$$\boldsymbol{x}(t) = h(\theta_1(t), \theta_2(t), \ldots, \theta_M(t)), \tag{1.2}$$

where $\boldsymbol{x}(t)$ is a function of movement parameters $\theta_1(t), \ldots, \theta_M(t)$. Notably, Churchland and colleagues showed that monkey motor cortex activity is better described by equations of the form Equation (1.1) than that of Equation (1.2) during a well-studied delayed reaching task [21, 166]. In this task, monkeys are trained to perform centre-out reaches on a fronto-parallel screen. At the beginning of each trial, monkeys hold their hands at the centre of the screen for a fixed duration. Next, a target appears on the screen, followed by a variable delay period ($0 - 1000$ ms), during which the animal is trained to stay still. At the end of the delay period, a go cue is given, signalling the monkeys to reach towards the target. Churchland et al. showed that during the "movement-epoch" (i.e. the period during which the reach unfolds), the population response in the motor cortex contains a large oscillatory component that is better explained by a dynamical systems perspective than by a representational one (see 118 for further comparisons of the two perspectives).

Another important piece of evidence for the dynamical systems view is that population activity converges to movement-specific subspaces during the delay period. Deviations of the population from these movement-specific subspaces at the go cue lead to longer reaction times [1], suggesting that a time-consuming preparatory process is needed to carefully seed the state of the motor cortex dynamical system into some movement-specific optimal subspace prior to movement onset. After the state of the motor cortex reaches the optimal subspace, it evolves away from it according to Equation (1.1), producing the desired muscle commands $\boldsymbol{m}(t)$ and reaching movement.

Lara et al. tested the necessity of such a time-consuming preparatory process for a range of movement conditions [95]. They found that such a preparatory process always precedes movement, irrespective of the length of the delay period, or whether the movement is self-paced or prompted by a go cue. Moreover, they found that the preparatory process can be fast (~ 150 ms), indeed much faster than the duration of the movement itself (~ 400 ms).

A theory of the motor cortex consistent with a decade of monkey electrophysiological experiments and analysis thus emerges. It suggests that (1) the motor cortex is a dynamical system, whose patterns of activity directly actuate the musculature and (2) a preparatory process progressively

seeds the movement-generating dynamics of the motor cortex with an appropriate initial condition prior to movement execution.

1.1.4 Controlling the controller

The brain's ability to control the state of the motor cortex goes beyond movement preparation. Brain-computer interface (BCI) experiments show that animals have a remarkable ability to modify the activity patterns of the motor cortex [59, 155]. In these experiments, population recordings are used to actuate the motion of a cursor on a screen in real-time. The animal's task is to control the cursor towards target locations on the screen, akin to performing centre-out arm reaches. By changing the readout from the recorded neuronal activity to the motion of the cursor, experimentalists are able to change the specific patterns of activity that must be produced for solving the task. They found that animals are able to quickly learn to produce these new target patterns of activity when they lie within the so-called "intrinsic manifold" [155], the linear subspace spanned by the activity patterns commonly visited during natural conditions (e.g., passively watching the movement of a cursor a screen). These experiments suggest the existence of neural mechanisms to "control the controller", i.e. a mechanism to control the state of the motor cortex which in turn actuates the musculature.

One potential mechanism by which the state of the motor cortex may be controlled is by receiving external inputs from other brain areas (i.e., $\boldsymbol{u}$ in Equation (1.1)). Guo et al. showed that input from the thalamus is necessary for maintaining persistent activity in the mouse anterior lateral motor cortex (ALM) during the delay epoch of a delayed licking task [63]. During this epoch, ALM activity also converges into lick-specific subspaces, similar to how motor cortex activity converges into reach-specific subspaces during the delay period of monkeys performing a delayed reaching task [98, 179].

Guo et al. also showed that the thalamus receives inputs from the ALM during the delay epoch, forming a thalamo-cortical loop [63]. In fact, Gao et al. later showed that in addition to the thalamo-cortical loop, there exists a cortico-cerebellar loop that is also important during the delay epoch of the same task. Perturbing these recurrent loops via photoinhibition (e.g., suppressing the activity of ALM-projecting neurons in the thalamus) disrupts the convergence of ALM activity into lick-specific subspaces during the delay epoch. What is more, they showed that suppressing ALM activity removes selectivity in the ALM-projecting neurons in thalamic and cerebellar neural populations. These findings suggest that thalamus and the cerebellum may be controllers of the motor cortex during the delay period, providing the necessary (feedback)

inputs for steering the state of the cortex to a desired subspace.

1.1.5 A model of optimal movement preparation and generation

In line with the idea that the motor cortex is under the active control of other brain circuits, models of the monkey motor cortex during the monkey delayed reaching task typically assume that the cortex receives some spatially-constant input during the preparatory period that sets its initial state prior to movement [73, 178]. These models are able to capture key aspects of the movement-generating dynamics observed in monkey motor cortex (e.g., multiphasic activity with a strong rotational component). However, these models do not capture the observed relationship between preparatory and movement-related neuronal activity, namely they evolve in largely orthogonal subspaces [37]. That is, the subspace that captures most of the activity variance during the delay epoch captures little variance during the movement epoch, and vice versa. Perhaps more importantly, these models do not address some of the outstanding theoretical questions concerning the dynamical systems perspective of the motor cortex: what is the optimal subspace that activity converges into during the delayed epoch? In what sense is this subspace optimal?

One of the primary contributions of this thesis is a model of movement preparation and generation that addresses some of the outstanding questions in the prevailing theory of the motor cortex dynamics described above. This model, described in Chapter 3, is inspired by the idea that the motor cortex is under the active control of another brain area (e.g., thalamus). We bring the methodology of optimal control — which has successfully explained behaviour [159, 185] — "inside the brain", proposing a perspective from which the motor cortex is the *plant* to be optimally controlled by other neural circuits [85]. Specifically, we posit that the state of the motor cortex is optimally prepared towards target initial states, in anticipation of the prospective movement. This corresponds to minimising — as quickly as possible — *the prospective motor error*, a measure which at any time prior to movement quantifies how much movement error would occur if the preparation process were to end and the movement triggered. This measure allows us to define the notion of an optimal subspace and derive an optimal control strategy during the preparatory process. The optimal strategy is feedback control, which we show could be achieved via a circuit model of the thalamo-cortical loop that respects known biological constraints such as Dale's law and known connectivity patterns from the cortex to the thalamus and from the thalamus to the cortex. Importantly, we show that optimal feedback control of the motor cortex during the delay period enables fast movement preparation, indeed preparation faster than the

duration of the reaching movement [95]. Moreover, optimal feedback control is necessary for the observed orthogonality between delay and movement epoch motor cortex dynamics [37].

Some of the essential control-theoretic tools used in Chapter 3 to derive the optimal control strategy such as observability and controllability are reviewed in Chapter 2. In particular, we discuss how these concepts can be and have been used in the literature to analyse the behaviour of biological and artificial neural circuits.

To construct the realistic circuit model in Chapter 3, we optimised the connectivity of the thalamocortical loop by differentiating through a specific algebraic matrix equation. In recent years, automatic differentiation has become the workhorse of most machine learning models that require large-scale gradient-based optimisations. Thus in Chapter 4 — as a standalone technical note — we derive the forward and reverse-mode gradients of several algebraic matrix equations commonly used in control theory.

1.2 Technical advances for analysing neural and behavioural recordings

Neuronal and behavioural recordings provide important constraints on theories of biological motor control. After all, the success of a theory depends on how well it explains experimental observations and generates new testable predictions. While data analysis provides insights that can be used to modify and update those theories, theories generate new predictions that can be tested in data, thus completing an *experiment-analysis-theory* cycle [133].

The experiment-analysis-theory cycle has been accelerated in recent years thanks in part to two factors. First, advances in neural recording technologies (e.g., calcium imaging, 202; NeuroPixels, 82) have enabled simultaneous recordings of thousands of neurons, producing data at a rate of terabytes per hour [133]. Not only have the number of simultaneously recorded neurons increased, the movement tasks have also become more complex, including multiple movement patterns and changes in reward structure. The advent of targeted neural perturbation technologies such as photoinhibition adds to the complexity of the task structure of the neural recordings. The availability of such large-scale, heterogeneous recordings allows neuroscientists to ask new questions that they would not have been able to ask otherwise.

Second, the exponential increase in the number of new machine learning methods for analysing large datasets over the last ten years has percolated into neuroscience, inspiring numerous new methods and techniques for behavioural and neural data analysis. For example, labelling

animal body positions in video recordings has traditionally been a labour-intensive process that required long hours of manual labelling and has been a limiting factor in accelerating the experiment-analysis-theory cycle. In recent years, this limitation has been partially removed by new methods such as DeepLabCut [115], which enables automatic labelling of body positions with deep neural networks.

As recording technology continues to improve in the future, there needs to be a commensurate, if not faster, improvement in scalable methods for analysing complex, large-scale experimental recordings. I now review supervised and unsupervised machine learning approaches to analysing neural recordings and relating them to behaviour. Then, I introduce one of the main contributions of this thesis along this direction: a method for identifying non-Euclidean latent variables from population recordings.

1.2.1 Supervised discovery of neural correlates of behaviour

Predicting behaviour from neural activity One approach to extracting information from neural recordings is to train supervised machine learning models to predict behaviour from neural recordings. Popular models include linear regression, kernel ridge regression, and deep neural networks. These methods allow us to identify the extent to which behaviourally-relevant information can be decoded from these recordings. For example, Stringer et al. used two-photon calcium imaging to record $\sim$10,000 neurons in the visual cortex of awake mice. They also recorded the facial behaviour of the mice during these recordings [172]. They showed that population activity is able to reliably predict the facial behaviour of the mice during both spontaneous behaviour and when the animal is passively viewing a stimulus, suggesting that visual cortex neurons encode both visual and behavioural information. Using NeuroPixels probes, Stringer et al. recorded thousands of neurons across multiple brain areas and showed that such behavioural information is also present in these brain areas. While such methods are unable to show that the brain uses the behavioural information for any computations (such conclusions can only be drawn through causal experiments such as photoinhibition experiments), these analyses still provide valuable insight into what information is present in different parts of the brain.

Predicting neural activity from behaviour Another way of applying supervised learning to neural data analysis is to reverse the direction of prediction: instead of predicting task-relevant variables from neural recordings, predict the activity of neurons based on these variables. One popular family of probabilistic models of spiking activity are Generalised Linear Models (GLMs; 134). A GLM predicts the spiking activity of a single neuron based on the history of (1) the spike trains of the population and (2) task-relevant variables such as behaviour and visual input. By dissecting the parameters of a trained GLM, we can identify the relative *importance* of different behavioural variables, and of the activity of the population itself, for predicting the spiking activity of each neuron. This can be viewed as a proxy for how "tuned" each neuron is to the task and the activity of other neurons.

Comparing artificial and biological networks trained on the same task Supervised learning is also used in an increasingly popular black-box, normative approach to modelling neural circuits and relating them to both neural recordings and behaviour [8, 211]. Specifically, the idea is to (i) train a neural network to perform the same task that an animal is trained to perform, (ii) analyse these trained agents to understand the algorithms and mechanisms that they use to solve the task, and (iii) compare the activity of the trained networks with neural recordings of animals performing the same motor task as the artificial agents. The hypothesis underlying this approach is that neural networks, biological or artificial, may find similar "solutions" when trained to perform the same task. By "opening the black-box", i.e. dissecting the trained networks and relating them to neural recordings, it may be possible to gain mechanistic insights about how the brain and the trained networks solve the task. This hypothesis has been partially validated in several studies that show that trained networks activity are similar to that of the neural recordings in a variety of tasks, including visual object recognition [211], decision making [110], time estimation [145], and motor control [100, 117, 119, 178, 184, 216]. By analysing the trained networks and comparing them with neural recordings, these studies have generate new hypotheses for how the brain solves each of these tasks.

In the motor domain, this black-box approach typically involves training a RNN to either generated recorded motor outputs (e.g., electromyograms; 178, 216) or to actuate a model of a body part to perform a specific task (e.g., outputs to muscles that actuate a two-link arm to produce reaching movements) [100, 117, 119, 184]. Detailed models of body biomechanics are made possible by physics-based simulators such as MuJoCo [186]. In addition to body biomechanics, there are also simulators that model the details of muscular contractions and force production (e.g., OpenSim; 30).

1.2.2 Unsupervised discovery of task-relevant latent variables

Latent variables models (LVMs) are widely applied for unsupervised neural data analysis. An LVM reduces the dimensionality of the datasets by identifying "latent variables", which best explain the dataset according to (1) some user-defined criterion (e.g., reconstruction of the dataset) as well as (2) prior knowledge about the dataset (e.g., temporal smoothness of the latent variables). For example, principal component analysis of neural recordings may be viewed as an LVM that extracts latent variables which capture most of the activity variance.

There are several reasons for the popularity of LVMs in neural data analysis. First, by reducing the dimensionality of the data, it is useful for visualising large data sets. For example, Afshar et al. used Gaussian Process Factor Analysis [213] to uncover single-trial latent trajectories from population recordings of the motor cortex during the delayed reaching task [1]. By plotting these latent-trajectories in three-dimensional space, we can visualise how single-trial activity converges to movement specific subspaces during the delay period and subsequently evolves according to the dynamics of the motor cortex during the movement epoch.

LVMs are also useful for comparing neural recordings to circuit models that are designed to model those neural recordings. For example, Sussillo et al. used principal component analysis to first extract low-dimensional latent trajectories in both monkey M1 recordings and an RNN trained to model those recordings [178]. Then, they applied canonical correlation analysis to the latent trajectories so as to compare the latent dynamical structure of the the recordings and the RNN.

By summarising information scattered across multiple trials, neurons, and behavioural conditions with *latent variables*, LVMs capture the "essence" of the neuronal recordings. For example, Latent Factor Analysis via Dynamical Systems (LFADS) is a LVM that is able to encode entire neural recordings in a distribution over the initial states of a recurrent neural network; it can even do this over multiple sessions, with some neurons missing across different sessions. By pooling information across multiple sessions, LFADs is able to reach state of the art performance predicting trial-by-trial behaviour from inferred neural activity.

LVMs have also been applied to infer neuronal tuning curves. For example, Wu et al. applied a Gaussian Process Latent Variable Model (GPLVM) to neural data analysis. While methods such as probabilistic PCA and GPFA assume linear tuning functions (i.e., linear mapping from the latent variables to the activity of each neuron), GPLVMs place a Gaussian Process prior on the tuning functions. This enables learning of highly non-linear tuning curves that capture the complex neuronal responses to the latent variables.

1.2.3 Manifold Gaussian Process Latent Variable Model

Most of the LVMs that are currently applied to neural data analysis assume a Euclidean latent space. However, movement kinematics likely evolve on some low-dimensional non-Euclidean manifold. For example, head rotations are naturally represented as elements in SO(3) and the configuration of a two link arm is naturally represented on a torus. Therefore, one of the important contributions of this thesis is an extension of Gaussian Process Latent Variable Models [97] to non-Euclidean latent spaces. Leveraging recent advances in machine learning that generalise the reparameterisation trick to Lie groups [41], we develop a variational inference algorithm for jointly inferring nonparametric tuning curves and latent variables that live on non-Euclidean spaces. We call our method Manifold Gaussian Process Latent Variable Model (mGPLVM). We verify the validity of our method on toy datasets and show that our method recovers latent ring topologies in neural recordings of the fly and mouse head direction systems.

As presented in Chapter 5, mGPLVM assumes a Gaussian noise model and an independent and identically distributed prior over the latent states. However, it can also be extended to incorporate non-Gaussian noise models (e.g., Poisson or Negative Binomial noise models) as well as smooth priors over the latent variables. At the time of writing, we are extending mGPVLM in these directions and implementing it as a general software package for fitting a variety of LVMs commonly used for neural data analysis.

Chapter 2

Neuroscience out of control: control-theoretic perspectives on neural circuit dynamics

Summary

This chapter presents the following article:

> Neuroscience out of control: control-theoretic perspectives on neural circuit dynamics. **Ta-Chu Kao** and Guillaume Hennequin (2019). *Current Opinion in Neurobiology*, 58, 122–129.

A major challenge in systems neuroscience is to understand how the dynamics of neural circuits give rise to behaviour. Analysis of complex dynamical systems is also at the heart of control engineering, where it is central to the design of robust control strategies. Although a rich engineering literature has grown over decades to facilitate the analysis of such systems, little of it has percolated into neuroscience so far. Here, we give a brief introduction to a number of core control-theoretic concepts that provide useful perspectives on neural circuit dynamics. We introduce important mathematical tools related to these concepts, and establish connections to neural circuit analysis, focusing on a number of themes that have arisen from the modern "state-space" view on neural population dynamics.

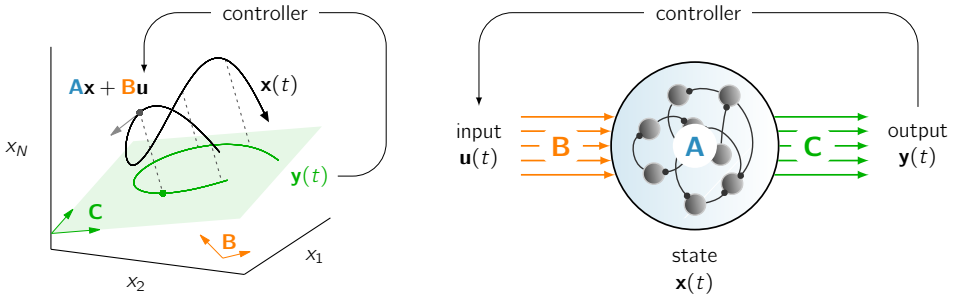


Figure 2.1: **State space & control perspective on neural dynamics.** The population activity vector $\mathbf{x}(t)$ traces out trajectories in state space (solid black, left diagram), following a flow (gray arrow) determined by the state matrix $\mathbf{A}$ as well as external inputs $\mathbf{u}(t)$ (right diagram; cf. Equation (2.1)). In a standard feedback control scenario, inputs are computed based on some measurements $\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t)$ (green) of the state vector, modifying the flow of activity along a few select "input channels" $\mathbf{B}$ (orange).

2.1 Introduction

Behaviour arises from dynamics that unfold in recurrently connected neural circuits, both locally within specialized regions and globally across multiple brain areas. To understand the principles that govern these dynamics, computational neuroscientists build model networks in a variety of ways: i) models that directly implement a number of known physiological features of the brain area(s) of interest [200], ii) low-dimensional latent dynamical models fitted to neural population recordings [21, 52, 131, 132], and iii) artificial neural networks trained to perform complex tasks [8]. In all cases, a critical research step is to understand the behaviour of the model through an analysis of its dynamics [174, 176].

Basic linear algebra provides useful geometric intuitions for the structure of population activity patterns, often represented as points in a so-called "state-space" (Figure 2.1; 46, 48, 154, 158, 166). Concepts such as projections, subspaces, nullspaces, etc, lie at the core of many linear dimensionality reduction techniques that are widely used to explore neural datasets [26, 27, 92], and help us reason geometrically about the computations carried out by neural circuits [59, 74, 87, 110, 138, 145, 155, 164, 178]. However, similar intuitions are more difficult to obtain for population *dynamics*, i.e. for the temporal evolution of neural activity in state space (Figure 2.1). This challenge is especially relevant given the recent shift in focus from static analyses to dynamical descriptions of circuit computations [21, 73, 110, 145, 166, 178]. Thus, an important goal in computational neuroscience is to develop an "algebra of dynamics", i.e. a set of conceptual, algebraic, and numerical tools for the study of neural circuits.

Control theorists have long been developing such tools [168]. Here, we review the key theoretical concepts of controllability, observability, model reduction and stability, which we find particularly relevant to the analysis of neural circuits. We briefly introduce these concepts in the control engineering context in which they were developed, and discuss the new perspectives that they provide on a number of recent results concerning the dynamics of neural computation.

2.2 State space models

A large part of classical control theory is dedicated to the control of linear state-space models of the form:

$$\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \text{noise} \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \text{noise}. \end{aligned} \tag{2.1}$$

Here, $\mathbf{A}$ is the so-called "state matrix" that governs the temporal evolution of the activity vector $\mathbf{x}(t)$ in the state space, possibly subject to process noise, and $\mathbf{B}$ is an "input matrix" whose columns define the directions along which time-varying inputs $\mathbf{u}(t)$ actuate the system (Figure 2.1). To control the state of the system (e.g. steer it along desired trajectories or towards desired end points), inputs are often derived as feedback from some (noisy) linear measurements $\mathbf{y}(t)$ of the full state, involving a readout matrix $\mathbf{C}$.

Equation (2.1) will look familiar to many neuroscientists. It lies at the heart of many statistical models, where it describes the dynamics of a set of "latent factors" that recapitulate the spatiotemporal structure in population recordings [13, 21, 151]. Due to their analytical tractability, linear(ized) models have also shed light on a variety of circuit computations, including short-term memory [50, 58, 203], selective amplification and surround suppression in primary visual cortex [121, 128], movement generation[73, 178], attention and decision-making [49], and probabilistic inference [2, 72]. Equation (2.1) has also been the workhorse of numerous studies on the origin, stimulus-dependence, and attentional-modulation of noise correlations [71, 75]. In many of these contexts, $\mathbf{x}(t)$ represents the vector of momentary firing rates in the network, $\mathbf{A}$ encapsulates recurrent connectivity and single-neuron leak, and $\mathbf{B}$ is a matrix of input synaptic weights.

Control theory textbooks contain a wealth of results that summarize important dynamical properties of systems governed by Equation (2.1), using matrices that are obtained through algebraic manipulations of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{C}$. We begin our review with two important such matrices that can be used to describe the input-output behaviour of a neural network: the observability and controllability Gramians.

2.3 Network observability

2.3.1 Control-theoretic context

Before even attempting to design a feedback control strategy, a control engineer will first establish feasibility. In particular, do the available partial observations $\mathbf{y}(t)$ of the full state $\mathbf{x}(t)$ provide enough information for efficiently controlling the system? Answers are found in the observability Gramian [168], a positive definite matrix associated with $\mathbf{A}$ and $\mathbf{C}$ and defined as

$$\mathbf{Q} = \int_0^\infty \exp\left(t\mathbf{A}^T\right) \mathbf{C}^T\mathbf{C} \exp\left(t\mathbf{A}\right) dt. \tag{2.2}$$

This matrix characterizes how well the initial state $\mathbf{x}(0)$, and therefore any subsequent state, can be inferred from observations of $\mathbf{y}(t)$ and $\mathbf{u}(t)$ – systems that make such inferences are often called "observers" [69]. Determining where $\mathbf{x}(0)$ was positioned along a given direction $\mathbf{v}$ in state space requires the quantity $\mathcal{E}(\mathbf{v}) = \mathbf{v}^T\mathbf{Q}\mathbf{v}$ to be strictly positive. If it is zero, one simply cannot resolve $\mathbf{x}(0)$ along $\mathbf{v}$, and such ambiguity makes controlling $\mathbf{x}$ more difficult. More generally, $\mathcal{E}(\mathbf{v})$ quantifies an ideal observer's confidence in this estimate. Conveniently, $\mathbf{Q}$ is also the solution to a simple "Lyapunov equation",

$$\mathbf{A}^T\mathbf{Q} + \mathbf{Q}\mathbf{A} + \mathbf{C}^T\mathbf{C} = 0, \tag{2.3}$$

with efficient solvers available for most programming languages.

2.3.2 Geometry of input sensitivity in neural circuits

The observability Gramian is a very useful tool for the analysis of neural circuits, as it provides information about a network's sensitivity to specific input patterns, or initial conditions. Specifically, $\mathcal{E}(\mathbf{v})$ above can be re-interpreted as the amount of "energy" in the output $\mathbf{y}(t)$ evoked by a pulse of input along direction $\mathbf{v}$ in state space [73] (Figure 2.2A and B). Input that momentarily pushes (or initializes) the state along a highly observable direction typically gives rise to strong and/or long output transients (Figure 2.2A, bottom left), especially in networks known as "nonnormal" which include all physiologically realistic models of brain circuits with balanced excitation and inhibition [58, 70, 73, 121]. In contrast, weakly observable initial conditions cause $\mathbf{y}(t)$ to decay away rapidly, or outright not to respond (Figure 2.2A, bottom right).

This connection was first exploited to examine the response properties of high-dimensional, inhibition-stabilized network models of primary motor cortex[73, 173]. It is also highly relevant to the current view of motor cortical networks as near-autonomous dynamical systems, where

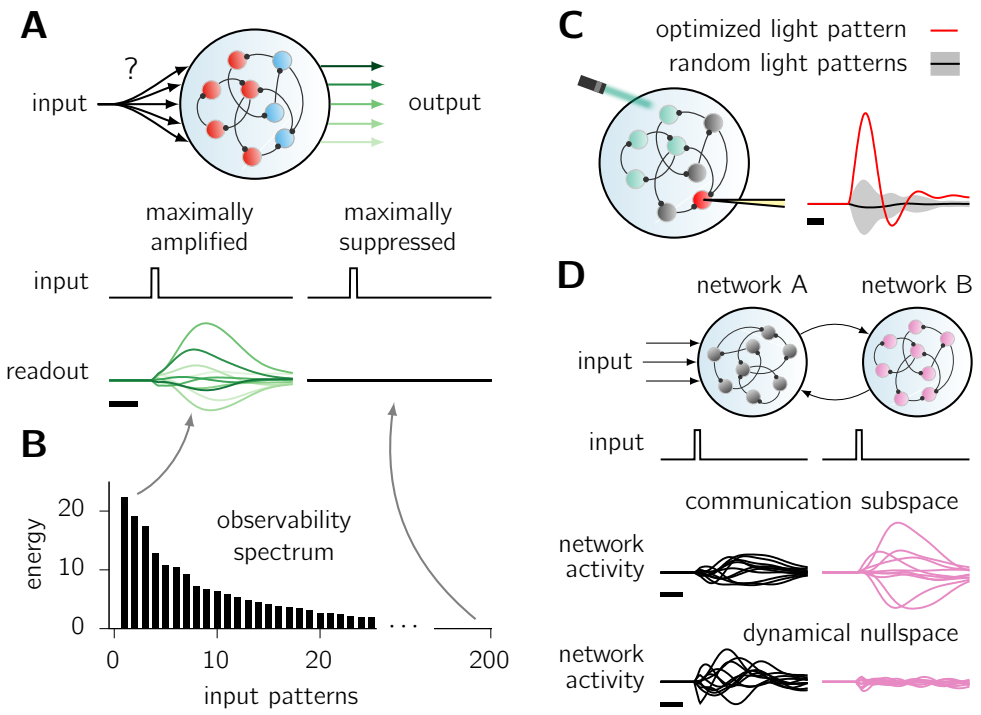


Figure 2.2: **Network observability.** **(A)** Interpretation of network observability as input sensitivity. Using simple algebraic techniques based on a neural network's observability Gramian **Q**, one can identify directions in state space (i.e. sets of input weights, black arrows) along which input pulses trigger large transients (bottom left) in the available readouts (shades of green), or, on the contrary, generate no output response at all (bottom right). **(B)** More generally, the (eigenbasis of the) observability Gramian defines a full set of orthogonal initial conditions that can be sorted by how much energy they evoke, from the most to the least. **(C)** Using the observability Gramian, one can determine the optimal way of stimulating a subset of neurons (green) to elicit the strongest possible response in another neuron (red). The optimal response (red) here is much larger than the typical response obtained through random stimulation patterns of the same energy (black; gray shading shows ± std.). **(D)** Illustration of communication subspaces and dynamical nullspaces, explained in the main text. Scale bars denote 20 ms in all panels.

initial conditions (a.k.a. preparatory states) largely determine the activity trajectories that follow [22, 137, 145, 166, 197]. The critical role of initial states is further corroborated by the success of modern data analysis methods such as LFADS [131], which use trial-specific initial conditions to accurately predict subsequent neural population activity through near-autonomous, nonlinear latent dynamics.

2.3.3 Optimal drive, robustness to perturbations

Knowing a circuit's sensitivity to inputs can lead to efficient strategies for interacting with it, e.g. to most efficiently drive or suppress some target cells. Figure 2.2C shows an example where the concept of observability is used to derive the optimal way of stimulating a subset of neurons, so as to elicit maximum response in another, non-stimulated neuron.

Conversely, emergent technologies for optical, random-access perturbations of neural dynamics [18, 38, 130] will soon enable the identification of observability spectra in neural circuits. This could be done using empirical techniques [65] similar to the classical reverse correlation-based mapping of receptive fields in sensory cortices. Alternatively, input sensitivity could be inferred by fitting dynamic latent-variable models, which — beyond explaining variance in the recorded data — incorporate the effect of external perturbations. Recently, by fitting such a model to monkey M1 recordings, Duncker et al. could explain the surprising robustness of M1 dynamics to optogenetic perturbations [33]. Their analysis suggested that optogenetic inputs transiently excite a set of weakly observable state-space directions, whose contributions to the momentary activity state tend to decay rapidly. This points to the existence of a "dynamical nullspace" in the circuit.

While weak observability can result from rapidly decaying directions in the full state space (as discussed above), it can also result from state-space trajectories evolving orthogonally to the subspace that defines the network readout (green plane in Figure 2.1, left). Such output-null dynamics may underlie movement preparation [87] or learning [138] in monkey, where preparatory cortical activity is largely orthogonal to movement-related activity [37]. An analogous phenomenon has also been observed in the premotor cortex of mice, where orthogonality seems achieved through anatomical segregation of preparatory neurons and output-related neurons. Indeed, this brain area contains a class of projection neurons that have a direct influence on movement, and which show little selectivity for the upcoming movement during preparation; these neurons are genetically distinct from other neurons that show strong preparatory selectivity, and are coupled to the thalamus [34].

2.3.4 Interacting brain areas

Beyond local circuits, dynamical nullspaces could also emerge from interactions between distant brain areas. For example, preparatory activity in mouse premotor cortex recovers from unilateral — but not bilateral — optical silencing during a delayed sensory discrimination task [98]. This suggests that it is the interactions between the two hemispheres that underpin the system's weak sensitivity to unilateral perturbation. More generally, input sensitivity may dictate how multiple, interconnected brain areas influence each other. Consider the two coupled networks of Figure 2.2D. Network B responds strongly when the activity of network A traverses their "communication subspace", i.e. when A's input to B spans B's most observable modes. On the other hand, it is also possible for network A to produce "private" activity fluctuations that do not influence B, so long as A's input falls in B's dynamical nullspace. A recent analysis of the joint dynamics of macaque areas V1 and V2 point to such signal propagation motifs [164]: a small "communication" subspace of V1 activity was found to predict V2 activity, with other "private" dimensions having little influence on V2 activity yet accounting for non-negligible variance in V1 activity. Mechanistically, selective propagation of signals across brain areas could — in theory — arise from a form of detailed excitation/inhibition balance, which only allows specific balance-breaking activity transients to propagate[73, 194].

2.4 Network controllability

2.4.1 Control-theoretic context

To establish control feasibility, an engineer will examine not only the observability of the system (c.f. above), but also its controllability. Can the state $\mathbf{x}(t)$ of the network be steered along any direction $\mathbf{v}$ in state space, using control inputs $\mathbf{u}(t)$ that can only actuate the network along a restricted set of directions (defined by the columns of $\mathbf{B}$; Figure 2.1)? Answers can be found in the controllability Gramian [168], a positive definite matrix associated with $\mathbf{A}$ and $\mathbf{B}$ and defined as:

$$\mathbf{P} = \int_0^\infty \exp\left(t\mathbf{A}\right) \mathbf{B}\mathbf{B}^T \exp\left(t\mathbf{A}^T\right) dt \tag{2.4}$$

(similar to the observability Gramian in Equation (2.2), $\mathbf{P}$ can be obtained by solving $\mathbf{A}\mathbf{P} + \mathbf{P}\mathbf{A}^T + \mathbf{B}\mathbf{B}^T = 0$). In particular, the minimum amount of input "energy" required to move the state of the system some distance along direction $\mathbf{v}$ is proportional to $\mathbf{v}^T\mathbf{P}^{-1}\mathbf{v}$; if this quantity is infinite, no clever control strategy will ever succeed in moving $\mathbf{x}$ along $\mathbf{v}$, whereas if it is small, control is easy and cheap. Figure 2.3A illustrates these ideas in a toy three-dimensional system.

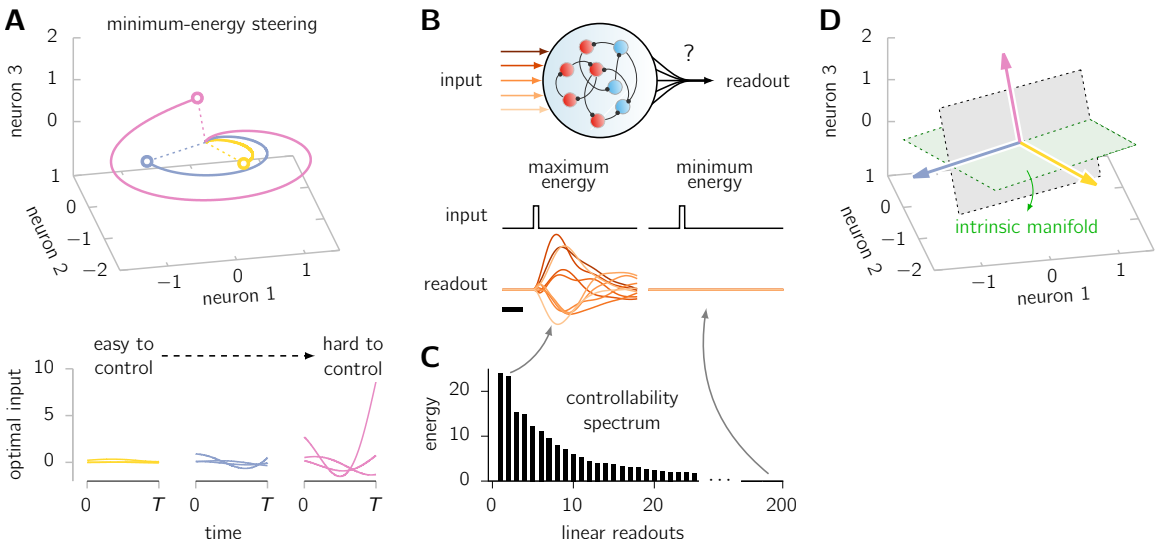


Figure 2.3: **Network controllability. (A)** Minimum-energy control of a 3-neuron network, which can be easily steered along two directions in state space (orange and blue), but much less easily along a third direction (pink). Solid lines (top) represent state-space trajectories under the action of optimal control inputs (bottom, same color code), ie. those of least "energy" that achieve the target end points (open circles) in finite time T. **(B)** Illustration of controllability for the same inhibition-stabilized network as used in Figure 2.2A [73]. The controllability Gramian contains information about the amount of variance (or "energy") in the activity of any readout, across all possible input channels (shades of orange) through which the network can be stimulated. In particular, it tells us which readout captures the most (bottom left) or least (right) state variance. The scale bar denotes 20 ms. **(C)** More generally, the (eigenbasis of the) controllability Gramian defines a full set of orthogonal state-space directions that can be sorted by how controllable they are, from the most to the least. **(D)** In BCI experiments, tasks that can be solved by modulating neural activity within the intrinsic manifold (green) can be learned more rapidly than tasks in which activity must span other, non-intrinsic directions (gray).

2.4.2 Intrinsic manifolds and control costs

The controllability Gramian $\mathbf{P}$ has a useful interpretation for the analysis of neural circuits: whereas the observability Gramian encodes a network's input sensitivity, $\mathbf{P}$ encodes the "intrinsic manifold" of the network's dynamics, i.e. the directions in state space that the network activity is most inclined to visit. This intrinsic manifold is a reflection of the network's connectivity (**A**) and input channels (**B**; Equation (2.4)). More formally, stimulating each of the available input channels individually results in a collection of state-space trajectories (Figure 2.3B) whose covariance matrix is, in fact, $\mathbf{P}$. Thus, the quantity $\sigma(\mathbf{v}) = \mathbf{v}^T\mathbf{P}\mathbf{v}$ is the average energy in these trajectories along state-space direction $\mathbf{v}$. In particular, directions with large $\sigma(\mathbf{v})$ (often referred to as "principal components") form the "natural repertoire" of the network, as they contribute a lot to state-space trajectories elicited by unspecific inputs (Figure 2.3B,C). In contrast, directions with small $\sigma(\mathbf{v})$ are almost never visited, unless the network is specifically driven by strong input patterns (Figure 2.3A).

The above connection between intrinsic manifold and control gives an interesting perspective on the results of recent brain-computer interface (BCI) experiments, in which monkeys modulate neural activity in a subset of M1 neurons to move a cursor on a screen [59, 74, 155]. As these studies show, how well a monkey can actuate a BCI strongly depends on the mapping from recorded M1 activity to BCI state variables (e.g. cursor velocity) — in other words, performance depends on the state-space directions that M1 activity needs to visit. In particular, monkeys struggle to learn new mappings that require M1 to produce activity outside its intrinsic manifold [155, 201]. In the toy illustration of Figure 2.3D, where the green plane depicts the intrinsic manifold, that would correspond to cursor velocity being suddenly associated with neural activity along the pink direction. In fact, even when the new mapping requires no such difficult excursions (for example, velocity originally defined by the yellow direction and now defined by the blue direction), the cloud of activity patterns generated within the intrinsic manifold under the new mapping appears similar to the one generated under the previous mapping [59, 74]. This suggests that learning occurs by repurposing a fixed distribution of within-manifold activity patterns, by reassociating each pattern with a new intended cursor movement.

These phenomena emerge naturally in networks such as the one discussed in Figure 2.3A, which has a set of highly controllable directions (forming the intrinsic manifold) and a set of poorly controllable ones (orthogonal to the intrinsic manifold). Modulation of neural activity within the intrinsic manifold is straightforward, as it can exploit the natural flow of activity without the need for large control inputs. In contrast, control of activity outside the intrinsic manifold requires

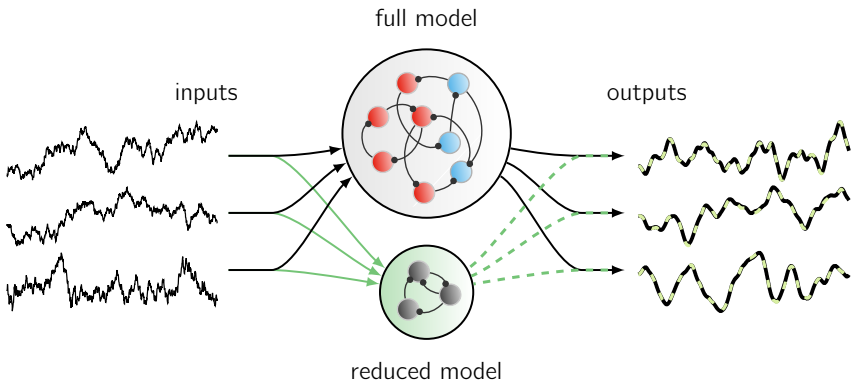


Figure 2.4: **Model reduction.** An inhibition-stabilized network with 200 neurons (from [73]) is reduced to a much smaller linear model of dimension 8. Here, the reduced model responds to time-varying inputs in almost exactly the same way as the full model does (compare black and dashed green outputs).

much larger inputs, perhaps larger than is physiologically feasible, thus limiting learnability of outside-manifold mappings. Moreover, if inputs to the network are energy-limited, controlling the state of the network along any direction within the intrinsic manifold will likely produce a fixed, common distribution of activity patterns that is determined by the controllability spectrum of the network [85].

2.4.3 Structural controllability

A recent study of the nervous system of C. elegans used an extension of the concept of controllability to investigate the role of various neuron classes in controlling the nematode's behaviour [212]. The authors considered the theoretical impact of individually ablating every possible class of neurons on the so-called "structural controllability" of the worm's muscles. They predicted that ablating certain classes of neurons would result in a reduction in the number of controllable muscles. Interestingly, while most of these neuron classes had already been causally implicated in the control of locomotion in previous studies, one neuron class had not been previously investigated. Ablation of these neurons resulted in small but significant motor impairments. Analogous predictions were also made and validated for the ablation of individual neurons within a class of motor neurons, illustrating the promise of applying control-theoretic methods to make causal predictions regarding the neural circuit basis of behaviour.

2.5 Model reduction

In control engineering, examination of observability and controllability often culminates in model reduction: the substitution of the original large-scale model with a more tractable, lower-dimensional model. Principled methods exist for performing dimensionality reduction while preserving the dynamic mapping from inputs to outputs. One such method is "balanced truncation" [168], whereby the state space is trimmed to eliminate directions that are both weakly controllable and weakly observable (Figure 2.4).

Critically, a reduced model is cheaper to simulate, enables computationally efficient control strategies, and is often easier to understand qualitatively. In neuroscience, model reduction has been applied to linearized models of signal propagation along active cables [47]. We speculate it will be useful for reverse engineering brain computations, by analyzing model networks that are either learned from data [131], or trained [8, 176] or hand-crafted [112] to perform a task. For example, trained models often need to be large enough to find solutions to their assigned task, but theses solutions often involve low-dimensional dynamics [110, 178]. Model reduction could help reveal and interpret these dynamics.

2.6 Stability and homeostasis

Control theory and neuroscience can also cross-fertilize in seeking to understand how complex, inherently unstable dynamical processes can be controlled and stabilized. Brain circuits contain major sources of dynamical instability that are kept in check by specific regulatory processes. Two prominent examples include i) positive feedback due to the presence of recurrent excitatory connections, which tend to cause runaway activity, and ii) positive feedback in Hebbian learning, causing runaway synaptic potentiation [214]. The control-theoretic methods aimed at stabilizing unstable systems could shed light on how neural circuits achieve stable behaviour. For example, even simple control rules such as homeostatic scaling of inhibitory synapses based on postsynaptic activity can restore E/I balance in networks with structural heterogeneity [94]. A more sophisticated form of Hebbian plasticity at inhibitory synapses [70] establishes and maintains stability in networks with strong excitatory feedback [195], and can be understood as an approximation to optimal robust stabilization algorithms [73].

On a more technical level, elements of systems theory have inspired useful parameterizations of (linear) latent dynamical models that guarantee stability, even when only limited data is available to constrain the system [13, 72]. Recently, systems theory has also been elegantly combined with

statistical approaches to characterize the stability and other aspects of the closed-loop behaviour of networks trained to have multiple fixed points [150].

2.7 Conclusions

The brain solves a variety of hard control problems, which are naturally studied in the framework used by engineers to tackle similar challenges. Control theory has long been applied to study motor control [159, 206], providing insights into the computational and algorithmic principles of internal models [60], error-corrective feedback [187], and planning [185]. Here, we have reviewed concepts that form the foundations of classical control theory and are specifically applicable to the analysis of neuronal network dynamics. They offer simple geometric descriptions of the dynamic input/output behaviour of neural circuits, along with a set of accessible analytical and numerical tools.

While the methods discussed here apply primarily to linear state-space models, they can be adapted to nonlinear systems in various ways, from standard linearization to more advanced operator-theoretic techniques [105]. Further extensions may be necessary for the analysis of recurrent neural networks that may be operating in highly nonlinear regimes. As we have noted, the observability and controllability Gramians afford several equivalent definitions in the linear case; nonlinear extensions will likely not capture all of them [65], and the specific needs of neuroscientists could in fact inspire the development of relevant extensions by the control community.

In reviewing the most basic properties of linear systems (i.e. those covered in the first few chapters of most control theory textbooks [168]), we have only scratched the surface. Control theory is primarily the science of feedback, and will continue to provide unique insights into how feedback is used in the brain to support the functions of single neurons, circuits, and systems. More generally, dissecting the circuit basis of complex computations such as motor control or reinforcement learning will benefit from a deeper integration of control theory, machine learning, and neuroscience.

Chapter 3

Optimal anticipatory control as a theory of motor preparation: a thalamo-cortical circuit model

Summary

This chapter presents the following article:

> Optimal anticipatory control as a theory of motor preparation: a thalamo-cortical circuit model. **Ta-Chu Kao**, Mahdieh S. Sadabadi and Guillaume Hennequin (2021). *Neuron* (in press).

Across a range of motor and cognitive tasks, cortical activity can be accurately described by low-dimensional dynamics unfolding from specific initial conditions on every trial. These "preparatory states" largely determine the subsequent evolution of both neural activity and behaviour, and their importance raises questions regarding how they are — or ought to be — set. Here, we formulate motor preparation as optimal anticipatory control of future movements, and show that the solution requires a form of internal feedback control of cortical circuit dynamics. In contrast to a simple feedforward strategy, feedback control enables fast movement preparation and predicts orthogonality between preparatory and movement activity, a distinctive feature of peri-movement activity in reaching monkeys. We propose a circuit model in which optimal preparatory control is implemented as a thalamo-cortical loop gated by the basal ganglia.

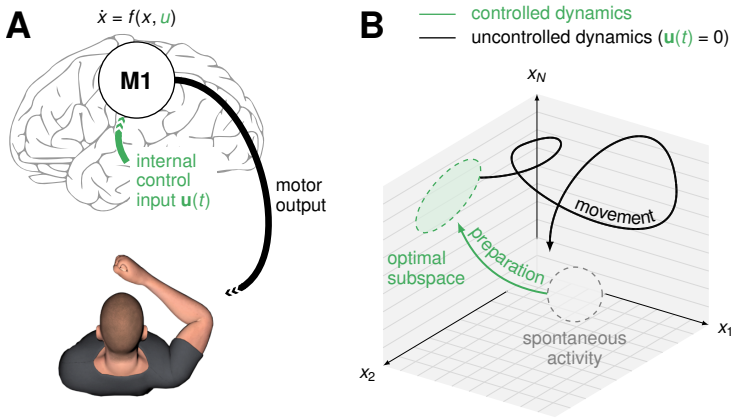


Figure 3.1: **Preparation & execution of ballistic movements. (A)** Under a dynamical systems view of motor control [166], movement is generated by M1 dynamics. Prior to movement, the M1 population activity state $\boldsymbol{x}(t)$ must be controlled into an optimal, movement-specific subspace in a phase of preparation; this requires internally generated control inputs $\boldsymbol{u}(t)$. **(B)** Schematic state space trajectory during movement preparation and execution.

3.1 Introduction

Fast ballistic movements (e.g. throwing) require spatially and temporally precise commands to the musculature. Many of these signals are thought to arise from internal dynamics in the primary motor cortex (M1; Figure 3.1A; 40, 127, 160, 166, 184). In turn, consistent with state trajectories produced by a dynamical system, M1 activity during movement depends strongly on the "initial condition" reached just before movement onset, and variability in initial condition predicts behavioural variability [1, 19, 131]. An immediate consequence of this dynamical systems view is the so-called "optimal subspace hypothesis" [22, 166]: the network dynamics that generate movement must be seeded with an appropriate initial condition prior to each movement. In other words, accurate movement production likely requires fine adjustment of M1 activity during a phase of movement preparation (Figure 3.1B, green).

The optimal subspace hypothesis helps to make sense of neural activity during the preparation epoch, yet several unknowns remain. What should the structure of the optimal preparatory subspace be? How does this structure depend on the dynamics of the cortical network during the movement epoch, and on downstream motor processes? Must preparatory activity converge to a single movement-specific state and be held there until movement initiation, or is some slack allowed? What are the dynamical processes and associated circuit mechanisms responsible for motor preparation? These questions can be (and have been partially) addressed empirically, e.g. through analyses of neural population recordings in reaching monkeys [3, 22, 37] or optogenetic

dissection of circuits involved in motor preparation [53, 63, 98, 157]. Yet, for lack of an appropriate theoretical scaffold, it has been difficult to interpret these experimental results within the broader computational context of motor control.

Here, we bridge this gap by considering motor preparation as an integral part of motor control. We show that optimal control theory, which has successfully explained behaviour [161, 187] and neural activity [100, 184] during the movement epoch, can also be brought to bear on motor preparation. Specifically, we argue that there is a prospective component of motor control that can be performed in anticipation of the movement (i.e. during preparation). This leads to a normative formulation of the optimal subspace hypothesis. Our theory specifies the control inputs that must be given to the movement-generating network during preparation to ensure that (i) any subsequent motor errors are kept minimal and (ii) movements can be initiated rapidly. These optimal inputs can be realized by a feedback loop onto the cortical network.

This normative model provides a core insight: the "optimal subspace" is likely high dimensional, with many different initial conditions giving rise to the same correct movement. This has an important consequence for preparatory control: at the population level, only a few components of preparatory activity impact future motor outputs, and it is these components only that need active controlling. By taking this into account, the optimal preparatory feedback loop dramatically improves upon a simpler feedforward strategy. This holds for multiple classes of network models trained to perform reaches, and whose movement-epoch dynamics are quantitatively similar those of monkey M1. Moreover, optimal feedback inputs, but not feedforward inputs, robustly orthogonalize preparatory- and movement-epoch activity, thus accounting for one of the most prominent features of perimovement activity in reaching monkeys [37, 87].

Finally, we propose a way in which neural circuits may implement optimal anticipatory control. In particular, we propose that cortex is actively controlled by thalamic feedback during motor preparation, with thalamic afferents providing the desired optimal control inputs. This is consistent with the causal role of thalamus in the preparation of directed licking in mice [63]. Moreover, we posit that the basal ganglia operate an ON/OFF switch on the thalamocortical loop [25, 66, 81, 103], thereby flexibly controlling the timing of both movement planning and initiation.

Beyond motor control, a broader set of cortical computations are also thought to rest on low dimensional circuit dynamics, with initial conditions largely determining behaviour [131, 169]. These computations, too, may hinge on careful preparation of the state of cortex in appropriate subspaces. Our framework, and control theory more generally, may provide a useful language for

reasoning about putative algorithms and neural mechanisms [86].

3.2 Results

3.2.1 A model of movement generation

We begin with an inhibition-stabilized network (ISN) model of the motor cortex in which a detailed balance of excitation and inhibition enables the production of rich, naturalistic activity transients (73, Figure 3.2A). This network serves as a pattern generator for the production of movement (we later investigate other movement-generating networks). Specifically, the network directly controls the two joint torques of a two-link arm (Section 3.4.1), via a linear readout of the momentary network firing rates:

$$\boldsymbol{m}(t) = \boldsymbol{C}\boldsymbol{r}(t). \tag{3.1}$$

Here, $\boldsymbol{m}(t)$ is a vector containing the momentary torques, and $\boldsymbol{r}(t)$ is the population firing rate vector (described below). The network has $N = 200$ neurons, whose momentary internal activations $\boldsymbol{x}(t) = (x_1, x_2, \ldots, x_N)^T$ evolve according to (29; Section 3.4.1):

$$\tau \frac{d\boldsymbol{x}}{dt} = -\boldsymbol{x}(t) + \boldsymbol{W}\,\boldsymbol{r}(t) + \overline{\boldsymbol{h}} + \boldsymbol{h}(t) \tag{3.2}$$

$$\boldsymbol{r}(t) = \phi[\boldsymbol{x}(t)]\,. \tag{3.3}$$

Here, τ is the single-neuron time constant, $\boldsymbol{W}$ is the synaptic connectivity matrix, and $\phi[x]$ (applied to $\boldsymbol{x}$ element-wise) is a rectified-linear activation function converting internal activations into momentary firing rates. The network is driven by two different inputs shared across all movements: a constant input $\overline{\boldsymbol{h}}$ responsible for maintaining a heterogeneous spontaneous activity state $\boldsymbol{x}_{\text{sp}}$, and a transient input $\boldsymbol{h}(t)$ arising at movement onset and decaying through movement. The latter input models the dominant, condition-independent timing-related component of monkey M1 activity during movement [88]. We note that, while the network model is generally nonlinear, it can be well approximated by a linear model ($\boldsymbol{r} = \boldsymbol{x}$) as only a small fraction of neurons are silent at any given time (Figure 3.12; Discussion). Our formal analyses here rely on linear approximations, but all simulations are based on Equations (3.2) and (3.3) with nonlinear ϕ.

We calibrated the model for the production of eight rapid straight reaches with bell-shaped velocity profiles (Figure 3.2B, top left; details in Section 3.4.1; see also Figure 3.11). To perform this calibration, we noted that – in line with the dynamical systems view of movement generation [166]

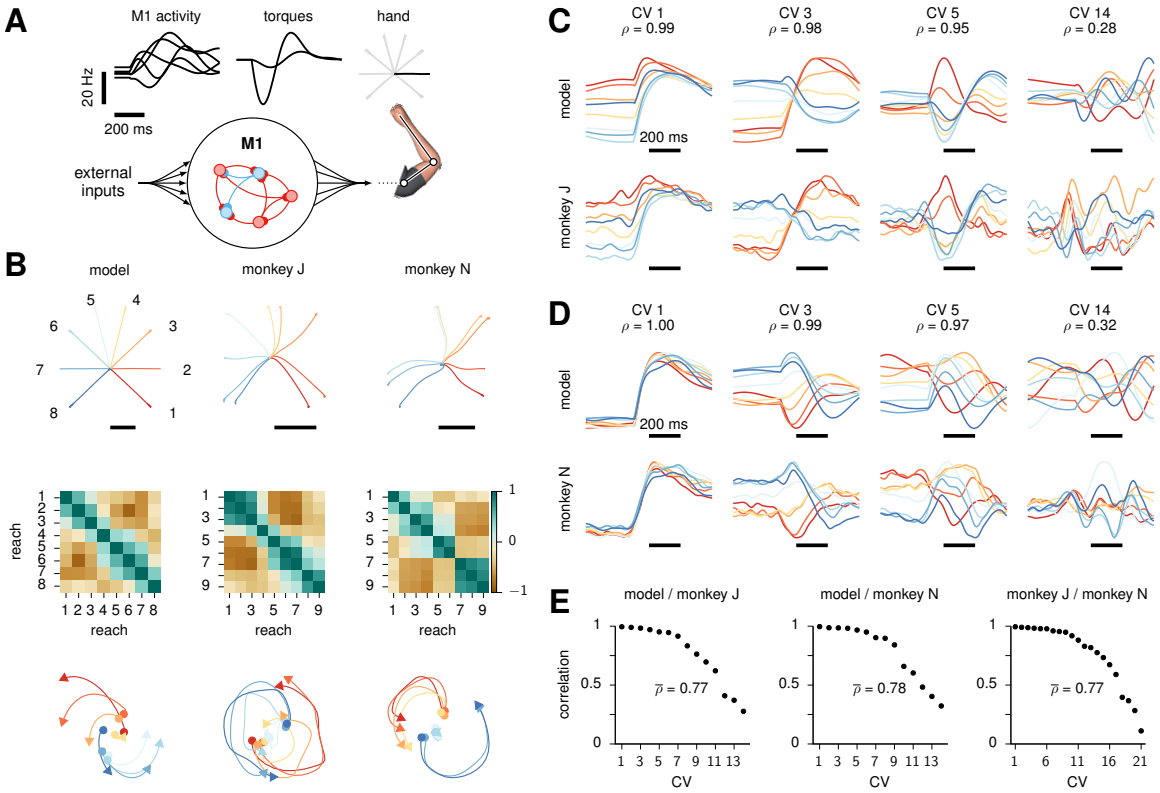


Figure 3.2: **Movement generation in a network model of M1 and comparison with monkey M1 dynamics during movement.** **(A)** Schematics of our M1 model of motor pattern generation. The dynamics of an excitation-inhibition network [73] unfold from movement-specific initial conditions, resulting in firing rate trajectories (left; 5 neurons shown) which are linearly read out into joint torques (middle), thereby producing hand movements (right). The model is calibrated for the production of eight straight center-out reaches; firing rates and torques are shown only for the movement colored in black. To help visualize initial conditions, firing rates are artificially clamped for the first 100 ms. **(B)** Top: model (left) and monkey (center and right) hand trajectories for each straight reach (color-coded). Black bars denote 10 cm. Monkey hand trajectories are averaged across trials with delays longer than 400 ms. Middle: overlap (Pearson correlation across neurons) between the preparatory end-states for model (left) and monkey activity (center and right). Reaches are numbered counter-clockwise as indicated near the model hand trajectories. Bottom: neural activity projected into the top jPC plane (see text). **(C)** Timecourse of the 1st, 3rd, 5th and 14th CCA projections (canonical variables) in the model (top) and monkey J (bottom), for each condition (color-coded as in B). Black scale bars indicate 200 ms from movement onset (note that "movement onset" in the model is re-defined to account for the latency between the go cue and actual movement onset in the monkey; see Section 3.5.2). To equalize the number of movement conditions across model and monkeys, we dropped the 9th movement, which is kinematically redundant the 8th (c.f. B). **(D)** Same as C, for monkey N (5th movement excluded, redundant with the 6th). **(E)** Full spectrum of canonical correlations, with average labeled $\bar{\rho}$.

– movements produced by our model depend strongly on the "initial condition", i.e. the cortical state $\boldsymbol{x}$ just before movement onset [1, 22]. We thus "inverted" the model numerically, by finding eight different initial conditions and a common readout matrix $\boldsymbol{C}$ such that the dynamics of the nonlinear model (Equations (3.2) and (3.3)), seeded with each initial condition, would produce the desired movement. Importantly, we constrained $\boldsymbol{C}$ so that its nullspace contained the network's spontaneous activity state, as well as all eight initial conditions. This constraint ensures that movement does not occur spontaneously nor during late preparatory stages when $\boldsymbol{x}(t)$ has converged to one of these initial conditions. Nevertheless, these constraints are not sufficient to completely silence the network's torque readout $\boldsymbol{m}(t)$ during preparation. While such spurious output tended to be very small in our simulations (Figure 3.10), the two stages of integration of $\boldsymbol{m}(t)$ by the arm's mechanics led to substantial drift of the hand before the reach. To prevent drift without modelling spinal reflexes for posture control, we artificially set $\boldsymbol{m}(t) = 0$ during movement preparation.

We re-analyzed population recordings of monkey M1/PMd during straight reaching (data courtesy of Mark Churchland, Matt Kaufman and Krishna Shenoy; 20). We found that our model captures several essential aspects of movement-related neural dynamics (Figure 3.2B-E). First, neurons exhibit heterogeneous, multiphasic oscillatory activity that often grows transiently from the preparatory end-state before shrinking back to spontaneous levels (Figure 3.2A; 73), similar to monkey M1 activity [21]. Second, kinematically similar reaches are produced from similar preparatory end-states in both model and monkeys (Figure 3.2B, middle). Third, jPCA [21] reveals consistent state-space rotations in peri-movement population activity in both model and monkeys (Figure 3.2B, bottom). Finally, canonical correlations analysis [178] uncovers subtantial overlap between monkey and model population activity across time and conditions (Figure 3.2C-D). Indeed, the average canonical correlation between the model and each of the two monkeys we analyzed is on a par with that between those two monkeys (Figure 3.2E).

3.2.2 Optimal control as a theory of motor preparation

Having calibrated our network model of movement generation, we now turn to preparatory dynamics. Shenoy et al.'s dynamical systems perspective suggests that accurate movement execution likely requires careful seeding of the generator's dynamics with an appropriate, reach-specific initial condition [1]. In our model, this means that the activity state $\boldsymbol{x}(t)$ of the cortical network must be steered towards the initial condition corresponding to the intended movement (Figure 3.1B, green). This process, which we call "preparatory control", forms the core of this

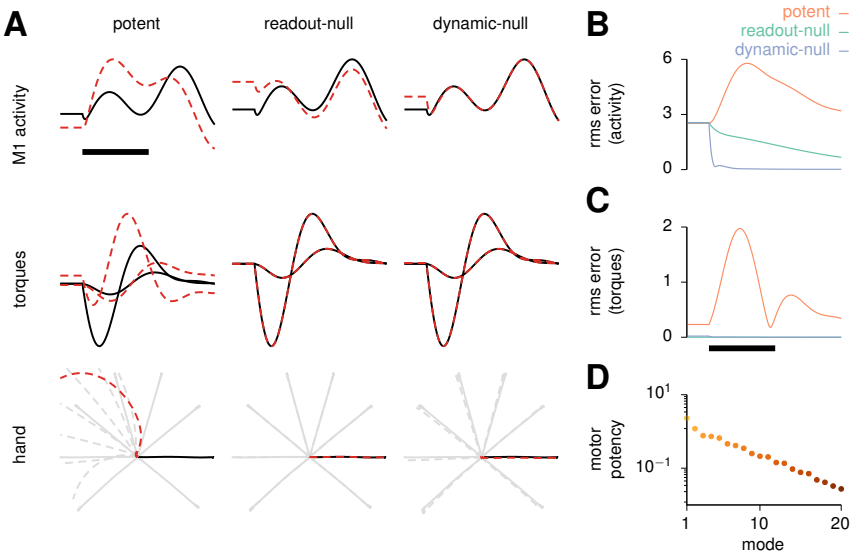


Figure 3.3: **Formalization of the optimal subspace hypothesis. (A)** Effect of three qualitatively different types of small perturbations of the initial condition (potent, readout-null, dynamic-null) on the three processing stages leading to movement (M1 activity,joint torques, and hand position), as already shown in Figure 3.2A. Unperturbed traces are shown as solid lines, perturbed ones as dashed red lines. Only one example neuron (top) is shown for clarity. Despite all having the same size here (Euclidean norm), these three types of perturbation on the initial state have very different consequences. Left: "potent" perturbations result in errors at every stage. Middle: "readout-null" perturbations cause sizeable changes in internal network activity but not in the torques. Right: "dynamic-null" perturbations are inconsequential at all stages. **(B)** Timecourse of root-mean-square error in M1 activity across neurons and reach conditions, for the three different types of perturbations. **(C)** Same as (B) but root-mean-square error in torques. **(D)** The motor potency of the top 20 most potent modes. In (A-C), signals are artificially held constant in the first 100 ms for visualization, and black scale bars denote 200 ms from movement onset.

study.

An important first step towards formalizing preparatory control and unravelling putative circuit mechanisms is to understand how deviations from "the right initial condition" impact the subsequent movement. Are some deviations worse than others? Mathematical analysis reveals that, depending on the direction in state space along which the deviation occurs, there may be strong motor consequences or none at all (Figure 3.3; Section 3.4.2). Some preparatory deviations are "prospectively potent": they propagate through the dynamics of the generator network during the movement epoch, modifying its activity trajectories, and eventually leading to errors in torques and hand motion (Figure 3.3A, left). Other preparatory deviations are "prospectively readout-null": they cause subsequent perturbations in cortical state trajectories, too, but these are correlated across neurons in such a way that they cancel in the readout and

leave the movement unaltered (Figure 3.3A, center). Yet other preparatory perturbations are "prospectively dynamic-null": they are outright rejected by the recurrent dynamics of the network, thus causing little impact on subsequent neuronal activity during movement, let alone on torques and hand motion (Figure 3.3A, right; Figure 3.3B-C).

We emphasize that the "prospective potency" of a preparatory deviation is distinct from its "immediate potency", i.e. from the direct effect such neural activity might have on the output torques [87, 196]. For example, the initial state for each movement is prospectively potent by construction, as it seeds the production of movement-generating network responses. However, it does not itself elicit movement on the instant, and so is immediately null. Having clarified this distinction, we will refer to prospectively potent/null directions simply as potent/null directions for succinctness.

The existence of readout-null and dynamic-null directions implies that, in fact, there is no such thing as "the right initial condition" for each movement. Rather, a multitude of initial conditions which differ along null directions give rise to the correct movement. To quantify the extent of this degeneracy, we measure the "prospective potency" of a direction by the integrated squared error in output torques induced during movement by a fixed-sized perturbation of the initial condition along that direction. We can then calculate a full basis of orthogonal directions ranking from most to least potent (an analytical solution exists for linear systems, Section 3.4.2, 86). For our model with only two readout torques, the effective dimensionality of the potent subspace is approximately 8 (Section 3.5.1; Figure 3.3D). This degeneracy substantially lightens the computational burden of preparatory ballistic control: there are only a few potent directions in state space along which cortical activity needs active controlling prior to movement initiation. Thus, taking into account the energetic cost of neural control, preparatory dynamics should aim at preferentially eliminating errors in preparatory states along those few directions that matter for movement.

We now formalize these insights in a normative model of preparatory motor control. We assume that, prior to movement, the initial condition for cortical dynamics is progressively reached during a preliminary phase of movement preparation. In this phase, the cortical network receives additional movement-specific control inputs $\boldsymbol{u}(t)$ (Figure 3.1, green) which are rapidly switched off to initiate movement:

$$\tau \frac{d\boldsymbol{x}}{dt} = -\boldsymbol{x}(t) + \boldsymbol{W}\,\boldsymbol{r}(t) + \overline{\boldsymbol{h}} + \boldsymbol{h}(t) + \boldsymbol{u}(t). \tag{3.4}$$

How should these preparatory inputs $\boldsymbol{u}(t)$ be chosen? At any time t during preparation, we can assign a "prospective motor error" $\mathcal{C}(\boldsymbol{x})$ to the current cortical state $\boldsymbol{x}(t)$. This prospective

error is the total error in movement that *would* result if movement was initiated at this time, i.e. if control inputs were suddenly switched off and the generator network was left to evolve dynamically from $\boldsymbol{x}(t)$ (Section 3.4.2). Note that $\mathcal{C}(\boldsymbol{x})$ is directly related to the measure of prospective potency described above. An ideal controller would supply the cortical network with such control inputs $\boldsymbol{u}(t)$ as necessary to lower the prospective motor error as fast as possible. This would enable accurate movement production in short order. We therefore propose the following cost functional:

$$\mathcal{J}[\boldsymbol{u}(t)] = \int_0^\infty \mathcal{C}(\boldsymbol{x}(t)) + \lambda \mathcal{R}(\boldsymbol{u}(t)) \; dt \tag{3.5}$$

where $\mathcal{R}(\boldsymbol{u})$ is an energetic cost which penalizes large control signals in excess of a baseline required to hold $\boldsymbol{x}(t)$ in the optimal subspace, and λ sets the relative importance of this energetic cost. Note that $\boldsymbol{x}(t)$ depends on $\boldsymbol{u}(t)$ via Equation (3.4).

3.2.3 Optimal preparatory control

When (i) the prospective motor error $\mathcal{C}$ is quadratic in the output torques $\boldsymbol{m}$, (ii) the energy cost $\mathcal{R}$ is quadratic in $\boldsymbol{u}$, and (iii) the network dynamics are linear, then minimizing Equation (3.5) corresponds to the well-known linear quadratic regulator (LQR) problem in control theory [168]. The optimal solution is a combination of a constant input and instantaneous (linear) state feedback,

$$\boldsymbol{u}_{\text{opt}}(t) = \boldsymbol{u}^\star + \boldsymbol{K}\, \delta\boldsymbol{x}(t), \tag{3.6}$$

where $\delta\boldsymbol{x}(t)$ is the momentary deviation of $\boldsymbol{x}$ from a valid initial condition for the desired movement (Section 3.4.2). In Equation (3.6), the constant input $\boldsymbol{u}^\star$ is movement specific, but the optimal gain matrix $\boldsymbol{K}$ is generic; both can be derived in algebraic form. Thus, even though the actual movement occurs in "open loop" (without corrective sensory feedback), optimal movement preparation occurs in closed loop, with the state of the pattern generator being controlled via internal feedback in anticipation of the movement. Later in this Results section, we build a realistic circuit model in which optimal preparatory feedback (Equation (3.6)) is implemented as a realistic thalamo-cortical loop. Prior to that, we focus on the core predictions of the optimal control law as an algorithm, independent of its implementation.

Optimal control inputs (Equation (3.6)) lead to multiphasic preparatory dynamics in the cortical network (Figure 3.4A, top). Single-neuron responses separate across movement conditions in much the same way as they do in the monkey M1 data (Figure 3.4C), with a transient increase in total across-movement variance (Figure 3.4D).The prospective motor error decreases very

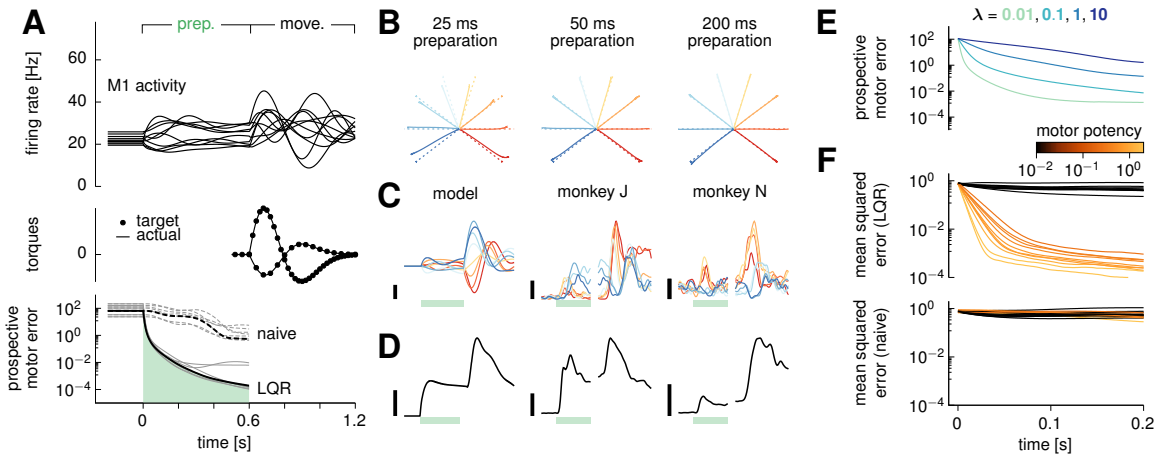


Figure 3.4: **Optimal preparatory control. (A)** Dynamics of the model during optimal preparation and execution of a straight reach at a 144-degree angle. Optimal control inputs are fed to the cortical network during preparation, and subsequently withdrawn to elicit movement. Top: firing rates of a selection of ten model neurons. Middle: generated torques (line), compared to targets (dots). Bottom: the prospective motor error $\mathcal{C}(\boldsymbol{x}(t))$ quantifies the accuracy of the movement if it were initiated at time t during the preparatory phase. Under the action of optimal control inputs, $\mathcal{C}(\boldsymbol{x}(t))$ decreases very fast, until it becomes small enough that an accurate movement can be triggered. The dashed line shows the evolution of the prospective cost for the naive feedforward strategy (see text). Gray lines denote the other 7 reaches for completeness. **(B)** Hand trajectories for each of the eight reaches (solid), following optimal preparation over a window of 25 ms (left), 50 ms (center) and 200 ms (right). Dashed lines show the target movements. **(C)** Firing rate of a representative neuron in the model (left), and the two monkeys (center and right) for each movement condition (color-coded as in Figure 3.2B). Green bars mark the 600 ms preparation window, black scale bars indicate 20 Hz. **(D)** Evolution of the average across-movement variance in single-neuron preparatory activity in the model (left) and the monkeys (center and right). Black scale bars indicate 16 Hz^2. **(E)** Prospective motor error during preparation, averaged over the eight reaches, for different values of the energy penalty parameter λ. **(F)** The state of the cortical network is artificially set to deviate randomly from the target movement-specific initial state at time $t = 0$, just prior to movement preparation. The temporal evolution of the squared Euclidean deviation from target (averaged over trials and movements) is decomposed into contributions from the 10 most and 10 least potent directions, color-coded by their motor potency as in Figure 3.3D. In (A-D) and (F), we used $\lambda = 0.1$.

quickly to negligible values (Figure 3.4A, bottom; note the small green area under the curve) as $\boldsymbol{x}(t)$ is driven into the appropriate subspace. After the preparatory feedback loop is switched off and movement begins, the system accurately produces the desired torques and hand trajectories (Figure 3.4A, middle). Indeed, movements are ready to be performed after as little as 50 ms of preparation (Figure 3.4B). We note, though, that it is possible to achieve arbitrarily fast preparation by decreasing the energy penalty factor λ in Equation (3.5) (Figure 3.4E). However, this is at the price of large energetic costs $\mathcal{R}(\boldsymbol{u})$, i.e. unrealistically large control inputs.

The neural trajectories under optimal preparatory control display a striking property, also observed in monkey M1 and PMd recordings [3, 95]: by the time movement is ready to be triggered (approx. 50 ms in the model), the firing rates of most neurons have not yet converged to the values they would attain after a longer preparation time — that is, $\|\delta\boldsymbol{x}(t)\| \gg 0$ (Figure 3.4A and C). Intuitively, this arises for the following reasons. First, the network reacts to the sudden onset of the preparatory input by following the flow of its internal dynamics, thus generating transient activity fluctuations. Second, the optimal controller is not required to suppress the null components of these fluctuations, which have negligible motor consequences. Instead, control inputs are used sparingly to steer the dynamics along potent directions only. Thus, the network becomes ready for movement initiation well before its activity has settled. To confirm this intuition, we artificially set $\boldsymbol{x}(t)$ at preparation onset to randomly and isotropically deviate from the target initial condition. We then computed the expected momentary squared deviation $\delta\boldsymbol{x}(t)$ along the spectrum of potent directions (shown in Figure 3.3D) during preparation. Errors are indeed selectively eliminated along directions with motor consequences, while they linger or even grow in other, inconsequential directions (Figure 3.4F, top).

Importantly, feedback control vastly outperforms a naive preparation strategy which uses a simpler, constant feedforward input $\boldsymbol{u}(t) = \boldsymbol{u}^\star$ and ignores the error feedback term ($\boldsymbol{K} = 0$ in Equation (3.6)). This strategy corresponds to the limit of zero input energy as it is defined in our framework (Section 3.4.2; it is also the optimal solution in the limit of $\lambda \to \infty$ in Equation (3.5)). Under this naive strategy, network activity successfully settles in a desired initial condition *eventually*, but the decrease in prospective motor error is much slower than under LQR (Figure 3.4A, bottom). Moreover, this decrease is non-selective, i.e. errors along potent and null directions are eliminated at the same rate (compare Figure 3.4F, top and bottom).

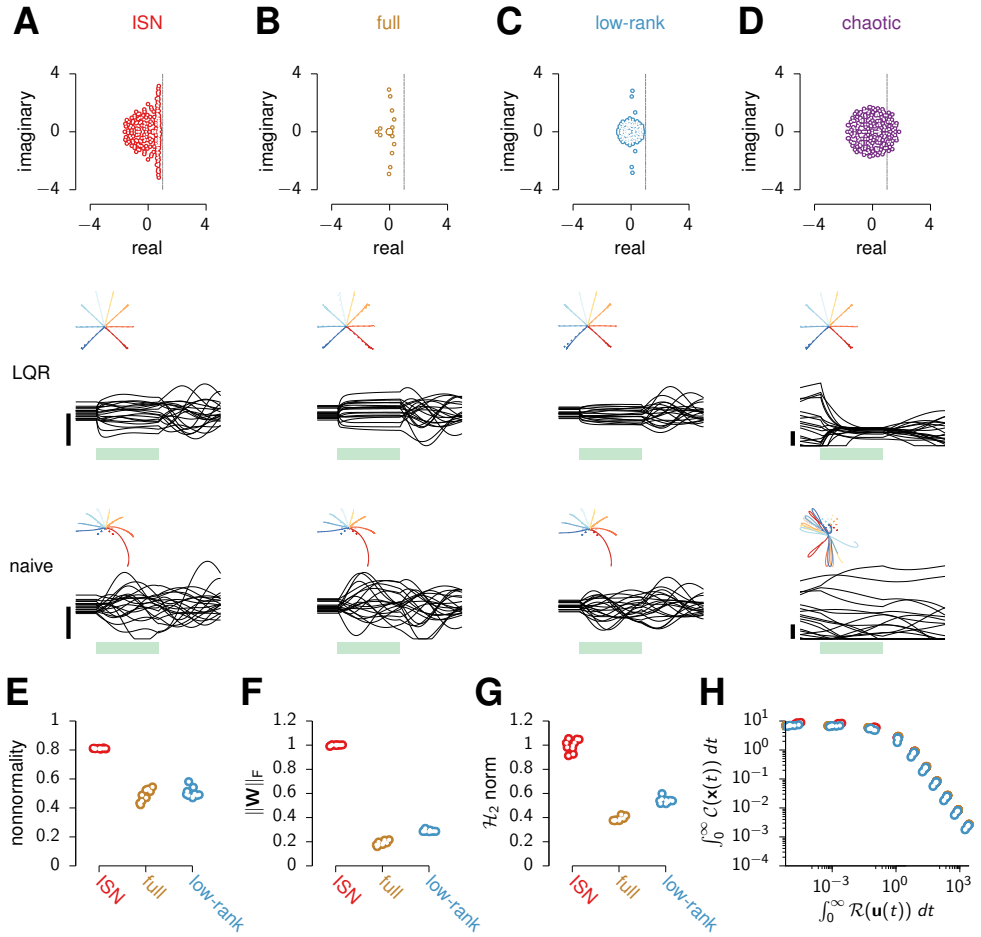


Figure 3.5: **Optimal preparatory control benefits other models of movement generation.** **(A)** ISN model. Top: eigenvalues of the connectivity matrix. Middle: activity of 20 example neurons during optimal preparatory control (LQR) and subsequent execution of one movement, with hand trajectories shown as an inset for all movements. The green bar marks the preparatory period. Bottom: same as middle, under the naive feedforward strategy. **(B-D)** Same as (A), for a representative instance of each of the three other network classes (see text). **(E-G)** Index of nonnormality (E), Frobenius norm of the connectivity matrix $\|\boldsymbol{W}\|_{\mathrm{F}}$ (F), and $\mathcal{H}_2$ norm (G) for the 10 network instances of each class (excluding chaotic networks for which these quantities are either undefined or uninterpretable). Dots are randomly jittered horizontally for better visualization. Both $\|\boldsymbol{W}\|_{\mathrm{F}}$ and the $\mathcal{H}_2$ norm are normalized by the ISN average. **(H)** Quantification of controllability in the various networks. Optimal control cost ($\int_0^\infty \mathcal{C}(\boldsymbol{x}(t))\,dt$ in Equation (3.5)) against associated energy cost ($\int_0^\infty \mathcal{R}(\boldsymbol{u}(t))\,dt$), for different values of the energy penalty parameter λ (varied implicitly). Note that the naive feedforward strategy corresponds to the limit of zero energy cost (y-"intercepts").

3.2.4 Preparatory control in other M1 models

So far we have shown that feedback control is essential for rapid movement preparation in a specific model architecture. The inhibition-stabilized network (ISN) model we have used (Figure 3.2A and Figure 3.5A) has strong internal dynamics, whose "nonnormal" nature (Figure 3.5E; 189)

gives rise to pronounced transient amplification of a large subspace of initial conditions [73]. This raises the concern that this ISN network might be unduly high-dimensional and difficult to control. After all, feedback control might not be essential in other models of movement generation for which the naive feedforward strategy might be good enough.

To address this concern, we implemented optimal control in two other classes of network models which we trained to produce eight straight reaches in the same way as we trained the ISN, by optimizing the readout weights and the initial conditions (Figure 3.5B and C; Section 3.4.1). Importantly, we also optimized aspects of the recurrent connectivity: either all recurrent connections ('full' networks, Figure 3.5B), or a low-rank parameterization thereof ('low-rank' networks, Figure 3.5C; 112, 175). We trained 10 instances of each network class, and also produced 10 new ISN networks for comparison. Empirically, we found that training was substantially impaired by the addition of the condition-independent movement-epoch input $\boldsymbol{h}(t)$ in Equation (3.2) in the full and low-rank networks. We therefore dispensed with this input in the training of all these networks, as well as in the newly-trained ISNs for fair comparison (the results presented below also hold for ISNs trained with $\boldsymbol{h}(t) \neq 0$).

The full and low-rank networks successfully produce the correct hand trajectories after training (Figure 3.5B and C, middle), relying on dynamics with oscillatory components qualitatively similar to the ISN's (eigenvalue spectra in Figure 3.5A-C, top). They capture essential aspects of movement-epoch population dynamics in monkey M1, to a similar degree as the ISN does (Figure 3.6). Nevertheless, the trained networks differ quantitatively from the ISN in ways that would seemingly make them easier to control. First, they are less non-normal (Figure 3.5E). Second, they have weaker internal dynamics than the ISN, as quantified by the average squared magnitude of their recurrent connections (Figure 3.5F). Third, these weaker dynamics translate into smaller impulse responses overall, as quantified by the $\mathcal{H}_2$ norm (Figure 3.5G; Section 3.5.1; 73, 86).

Although these quantitative differences suggest that optimal feedback control might be superfluous in the full and low-rank networks, we found this is not the case. In order to achieve a set prospective control performance, all networks require the same total input energy (Figure 3.5H). In particular, the feedforward strategy (limit of zero input energy) performs equally badly in all networks: preparation is unrealistically slow, with 200 ms of preparation still resulting in large reach distortions (Figure 3.5A-C, bottom). In fact, we were able to show formally that the performance of feedforward control depends only on the target torques specified by the task, but not on the details of how these targets are achieved through specific initial conditions, recurrent connectivity $\boldsymbol{W}$, and readout matrix $\boldsymbol{C}$ (Section 3.4.7). Stronger still, our derivations explain

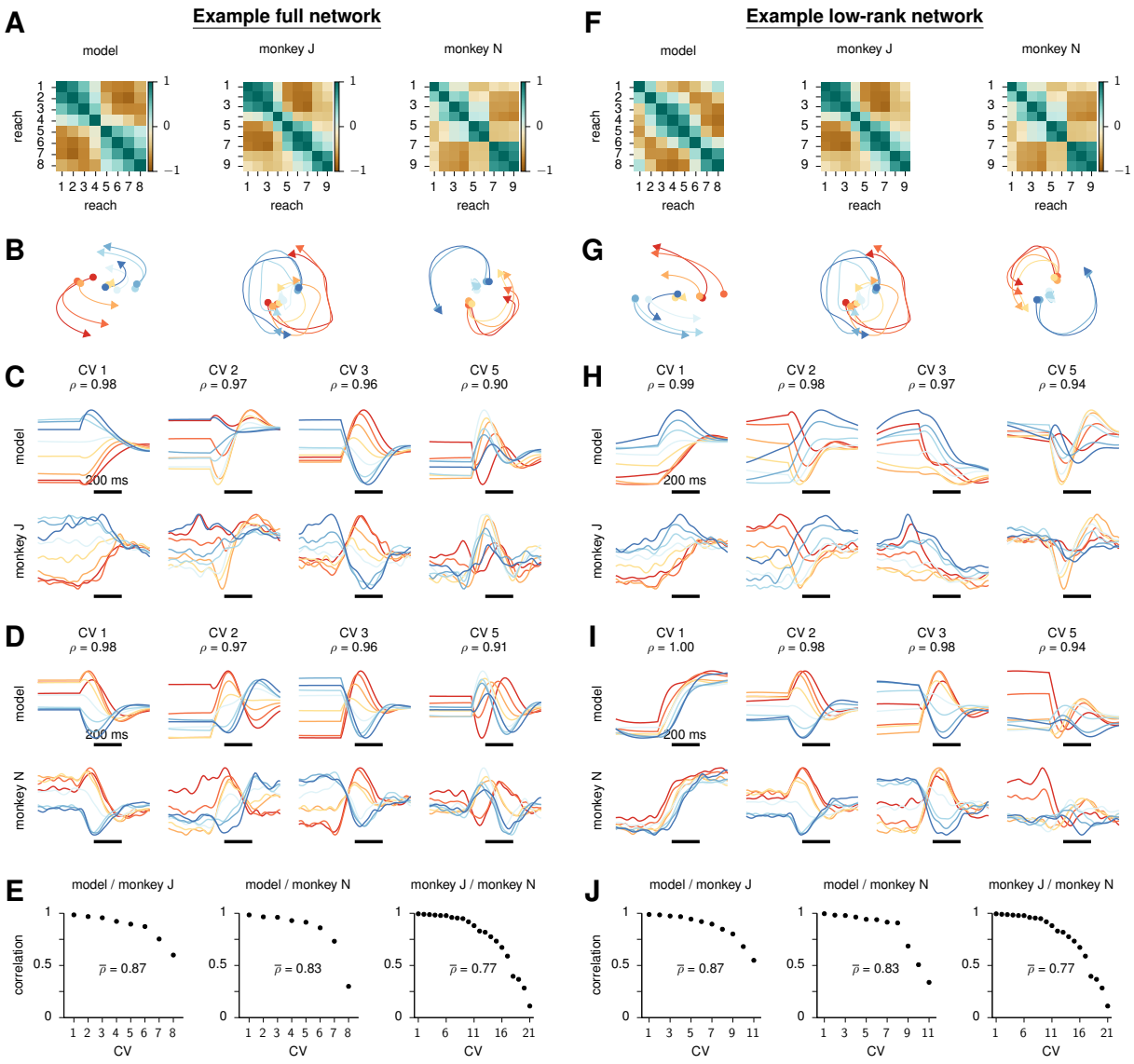


Figure 3.6: Movement-generating dynamics in an example 'full' network **(A-E)** and an example 'low-rank' network **(F-J)**, and comparison to monkey M1 dynamics. For details, please see the caption of Figure 3.2 where the same panels are shown for the ISN model.

why the movement errors resulting from insuffiently long feedforward preparation are identical in every detail across all networks (Figure 3.5A-C, bottom).

Finally, we also considered networks in the chaotic regime with fixed and strong random connection weights and the same threshold-linear activation function ('chaotic' networks, Figure 3.5D; 83, 111). In these networks, since static inputs are unable to quench chaos, the naive feedforward strategy cannot even establish a fixed point, let alone a correct one (Figure 3.5D, bottom). However, by adapting the optimal feedback control solution to the nonlinear case (Section 3.4.4), we found that it successfully quenches chaos during preparation and enables fast movement initiation (Figure 3.5D, middle).

In summary, the need for feedback control during preparation is not specific to our specific ISN model, but emerges broadly in networks of various types trained to produce reaches.

3.2.5 Orthogonal preparatory and movement subspaces

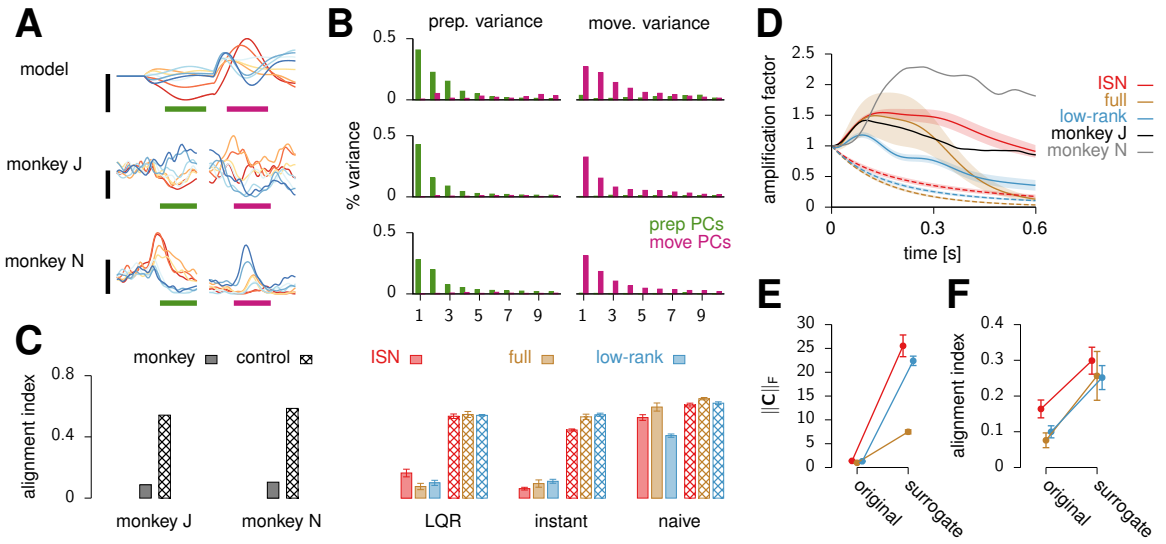


Figure 3.7: **Reorganization between preparatory and movement activity in model and monkey. (A)** Example single-neuron PSTHs in model (top) and monkey M1/PMd (middle and bottom), for each reach (c.f. Figure 3.2B). **(B)** Fraction of variance (time and conditions) explained during movement preparation (left) and execution (right) by principal components calculated from preparatory (green) and movement-related (magenta) trial-averaged activity (Section 3.5.2). Only the first 10 components are shown for each epoch. Variance is across reach conditions and time in 300 ms prep. and move. windows indicated by green and magenta bars in (A). The three rows correspond to those of (A). **(C)** Alignment index calculated as in Elsayed et al. [37], for the two monkeys (left) and the three classes of trained networks (colors as in Figure 3.5A-C) under three different preparation strategies (LQR, instant, naive; right). Here, preparation is long enough (500 ms) that even the naive feedforward strategy leads to the correct movements in all networks. Hashed bars show the average alignment index between random-but-constrained subspaces drawn as in Elsayed et al. [37] (see text and Section 3.5.2). **(D)** Amplification factor, quantifying the growth of the centered population activity vector $\boldsymbol{x}(t) - \boldsymbol{x}_{\text{sp}}$ during the course of movement, relative to the pre-movement state (Section 3.5.1). It is shown here for the two monkeys (black and gray), as well as for the three classes of trained networks (solid) and their surrogate counterparts (dashed). Shaded region denotes ± 1 s.t.d. around the mean across the 10 instances of each network class. To isolate the autonomous part of the movement-epoch dynamics, here we set $\boldsymbol{h}(t) = 0$ in Equation (3.2). **(E-F)** Frobenius norm $\|\boldsymbol{C}\|_{\text{F}}$ of the readout weights (F), and alignment index under the 'instant' preparation strategy (G; see text), for the original trained networks and their surrogates. Error bars denote ± 1 s.t.d. across the 10 networks of each class.

Optimal anticipatory control also accounts for a prominent feature of monkey motor cortex

responses during motor preparation and execution: across time and reach conditions, activity spans orthogonal subspaces during the two epochs. To show this, we followed Elsayed et al. and performed principal components analysis (PCA) on trial-averaged activity during the two epochs separately, in both the ISN model of Figure 3.4 and the two monkey datasets (Section 3.5.2; Figure 3.7A). We then examined the fraction of variance explained by both sets of principal components (prep-PCs and move-PCs) during each epoch. Activity in the preparatory and movement epochs are, respectively, approximately 4- and 6-dimensional for the model, 5- and 7-dimensional for monkey J, and 8- and 7-dimensional for monkey N (assessed by the "participation ratio"; Section 3.5.1).

Moreover, consistent with the monkey data, prep-PCs in the model account for most of the activity variance during preparation (by construction; Figure 3.7B, left), but account for little variance during movement (Figure 3.7B, right). Similarly, move-PCs capture little of the preparatory-epoch activity variance.

To systematically quantify this (lack of) overlap for all the trained models of Figure 3.5A-C and compare with monkey data, we used Elsayed et al.'s "alignment index". This is defined as the amount of preparatory-epoch activity variance captured by the top K move-PCs, further normalized by the maximum amount of variance that any K-dimensional subspace can capture. Here, K was chosen such that the top K prep-PCs capture 80% of activity variance during the preparatory-epoch. Both monkeys have a low alignment index (Figure 3.7C, left), much lower than a baseline expectation reflecting the neural activity covariance across both task epochs ("random" control in 37; Section 3.5.2). All trained models show the same effect under optimal preparatory control (Figure 3.7C, 'LQR'). Importantly, alignment indices arising from the naive feedforward solution are much higher and close to the random control.

To understand the origin of this effect, we first note that the preparatory end-states themselves tend to be orthogonal to movement-epoch activity in the models. Indeed, artificially clamping network activity to its end state during the whole preparatory epoch yields low alignment indices in all trained networks (Figure 3.7C, 'instant'). We hypothesize that this effect arises from the penalty on large readout weights introduced in the training procedure to protect from overfitting. This penalty encourages networks to find solutions whereby internal activity grows and decays during the course of movement as in the monkey data (Figure 3.7D), with an envelope reflecting that of the target torques. Since the autonomous (near-linear) dynamics that drive these activity transients are stable, the initial growth of activity at movement onset must be accompanied by a rotation away from the initial condition. Otherwise the growth would continue, contradicting stability.

To substantiate this interpretation, for each network that we trained, we built a surrogate network that achieved the same task without resorting to transient growth. We thus predicted higher alignment indices in these surrogate networks. To construct them, we noted that by applying a similarity transformation simultaneously to $\boldsymbol{W}$, $\boldsymbol{C}$ and the initial conditions (Section 3.4.1), one can arbitrarily suppress transient activity growth at movement onset yet maintain performance in the task in the linear regime (a similarity transformation does not change the overall transfer function from initial condition to output torques). In particular, the similarity transformation that diagonalizes $\boldsymbol{W}$ returns a purely normal network for which the magnitude of movement-epoch population activity can only decay during the course of movement, unlike that of the original network and that of the two monkeys (Figure 3.7D, dashed). Consistent with our hypothesis, these surrogate networks have larger readout weights (Figure 3.7E), and indeed a higher alignment index (Figure 3.7F).

So far, we have shown that orthogonality between *late*-preparatory and movement subspaces emerges generically in regularized trained networks. However, this does not fully explain why the alignment index is low under LQR in all models when early-/mid- preparatory activity is considered instead, as in Elsayed et al. [37]. In fact, under the naive feedforward strategy, early preparatory activity can be shown to be exactly the negative image of movement-epoch activity, up to a relatively small constant offset (Section 3.4.8). Thus, the temporal variations of early preparatory- and movement-epoch activity occur in a shared subspace, generically yielding a high alignment index (Figure 3.7C, 'naive'). In contrast, optimal control rapidly eliminates preparatory errors along potent directions, which contribute significantly to movement-epoch activity: if $\boldsymbol{x}(t)$ during movement had no potent component, by definition the movement would stop immediately. Thus, one expects optimal control to remove a substantial fraction of overlap between preparatory- and movement-epoch activity. What is more, by quenching temporal fluctuations along potent directions early during preparation, optimal feedback also suppresses the subsequent transient growth of activity that these potent fluctuations would have normally produced under the naive strategy (and which are indeed produced during movement). The combination of these primary and secondary suppressive effects results in the lower alignment index that we observe (Figure 3.7C, 'LQR').

In summary, orthogonality between preparatory and movement activity in monkey M1 is consistent with optimal feedback control theory. In the ISN, as well as in other architectures trained on the same task, orthogonality arises robustly from optimal preparatory control, but not from feedforward control.

3.2.6 Circuit model for preparatory control: a gated thalamocortical loop

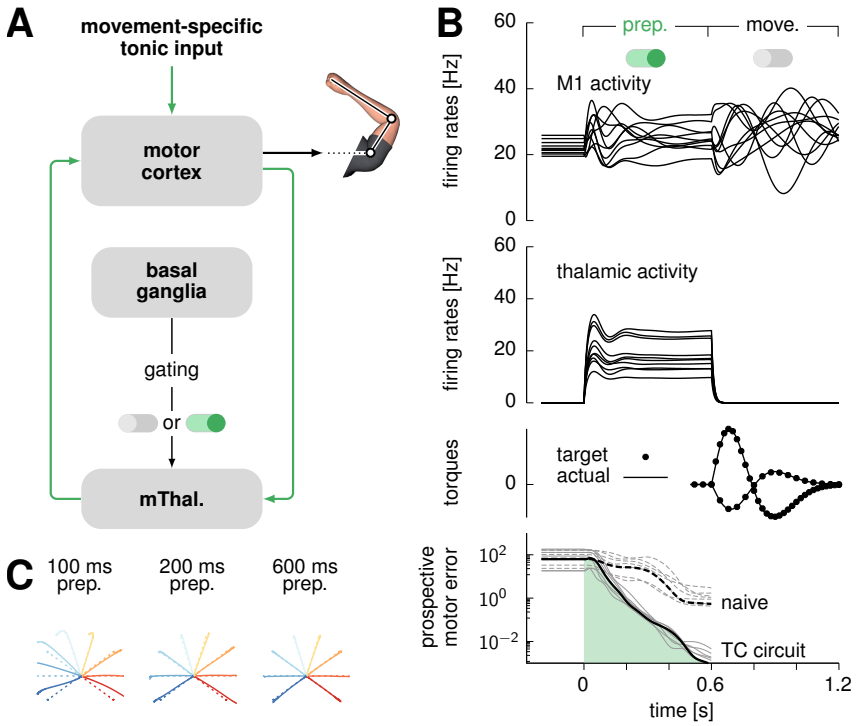


Figure 3.8: **Optimal movement preparation via a gated thalamo-cortical loop. (A)** Proposed circuit architecture for the optimal movement preparation (cf. text). **(B)** Cortical (top) and thalamic (upper middle) activity (10 example neurons), generated torques (lower middle), and prospective motor error (bottom) during the course of movement preparation and execution in the circuit architecture shown in (A). The prospective motor error for the naive strategy is shown as a dotted line as in Figure 3.4A. All black curves correspond to the same example movement (324-degree reach), and gray curves show the prospective motor error for the other 7 reaches. **(C)** Hand trajectories (solid) compared to target trajectories (dashed) for the eight reaches, triggered after 100 ms (left), 200 ms (middle) and 600 ms (right) of motor preparation.

So far we have not discussed the source of optimal preparatory inputs $\boldsymbol{u}(t)$, other than saying that they close a feedback loop from the cortex onto itself (Equation (3.6)). Such a loop could in principle be absorbed in local modifications of recurrent cortical connectivity [175]. However, we are not aware of any mechanism that could implement near-instant ON/OFF switching of a select subset of recurrent synapses, as required by the model at onset of preparation (ON) and movement (OFF). If, instead, the preparatory loop were to pass through another brain area, fast modulation of excitability in that relay area would provide a rapid and flexible switch [44]. We

therefore propose the circuit model shown in Figure 3.8A, where the motor thalamus acts as a relay station for cortical feedback [63, 123]. The loop is gated ON/OFF at preparation onset/offset by the (dis)-inhibitory action of basal ganglia outputs [25, 32, 66, 81, 103]. Specifically, cortical excitatory neurons project to 160 thalamic neurons, which make excitatory backprojections to a pool of 100 excitatory (E) and 100 inhibitory (I) neurons in cortex layer 4. In turn, these layer 4 neurons provide both excitation and inhibition to the main cortical network, thereby closing the control loop. Here, inhibition is necessary to capture the negative nature of optimal feedback. In addition to thalamic feedback, the cortical network also receives a movement-specific constant feedforward drive during preparation (analogous to $\boldsymbol{u}^\star$ in Equation (3.6) for the standard LQR algorithm; this could also come from the thalamus).

The detailed patterns of synaptic efficacies in the thalamo-cortical loop are obtained by solving the same control problem as above, based on the minimization of the cost functional in Equation (3.5) (mathematical details in Section 3.4.6). Importantly, the solution must now take into account some key biological constraints: (i) feedback must be based on the activity of the cortical E neurons only, (ii) thalamic and layer-4 neurons have intrinsic dynamics that introduce lag, and (iii) the sign of each connection is constrained by the nature of the presynaptic neuron (E or I).

The circuit model we have obtained meets the above constraints and enables flexible, near-optimal anticipatory control of the reaching movements (Figure 3.8B). Before movement preparation, thalamic neurons are silenced due to strong inhibition from basal ganglia outputs (not explicitly modelled), keeping the thalamocortical loop open (inactive) by default. At the onset of movement preparation, rapid and sustained disinhibition of thalamic neurons restores responsiveness to cortical inputs, thereby closing the control loop (ON/OFF switch in Figure 3.8B, top). This loop drives the cortical network into the appropriate preparatory subspace, rapidly reducing prospective motor errors as under optimal LQR feedback (Figure 3.8B, bottom). To trigger movement, the movement-specific tonic input to cortex is shut off, and the basal ganglia resume sustained inhibition of the thalamus. Thus, the loop re-opens, which sets off the uncontrolled dynamics of the cortical network from the right initial condition to produce the desired movement (Figure 3.8C).

The neural constraints placed on the feedback loop are a source of suboptimality with respect to the unconstrained LQR solution of Figure 3.4. Nevertheless, movement preparation by this thalamocortical circuit remains fast, on par with the shortest preparation times of primates in a quasi-automatic movement context [95] and much faster than the naive feedforward strategy (Figure 3.8B, bottom).

3.2.7 Model prediction: selective elimination of preparatory errors following optogenetic perturbations

An essential property of optimal preparatory control is the selective elimination of preparatory errors along prospectively potent directions. As a direct corollary, the model predicts selective recovery of activity along those same directions following preparatory perturbations of the dynamics, consistent with results by Li et al. [98] in the context of a delayed directional licking task in mice. Here, we spell out this prediction in the context of reaching movements in primates by simulating an experimental perturbation protocol similar to that of Li et al. [98], and by applying their analysis of population responses. These concrete predictions should soon become testable with the advent of optogenetics techniques in primates [129].

As our E/I cortical circuit model operates in the inhibition-stabilized regime [73, 128, 156, 190], we were able to use the same photoinhibition strategy as in Li et al. [98] to silence the cortical network (Figure 3.9A) in the thalamo-cortical circuit of Figure 3.8. We provided strong excitatory input to a random subset (60%) of inhibitory neurons, for a duration of 400 ms starting 400 ms after preparation onset. We found that "photoinhibition" has mixed effects on the targeted neurons: some are caused to fire at higher rates, but many are paradoxically suppressed (Figure 3.9B, top). For E cells and untargeted I cells, though, the effect is uniformly suppressive, as shown in Figure 3.9B (middle and bottom).

The perturbation transiently resets the prospective motor error to pre-preparation level, thus nullifying the benefits of the first 400 ms of preparation (Figure 3.9C). Following the end of the perturbation, the prospective motor error decreases again, recovering to unperturbed levels (Figure 3.9C, solid vs. dashed) and thus enabling accurate movement production (compare middle and bottom hand trajectories). This is due to the selective elimination of preparatory errors discussed earlier (Figure 3.4F): indeed, shuffling $\delta\boldsymbol{x}$ across neurons immediately prior to movement, thus uniformizing the distribution of errors in different state space directions, leads to impaired hand trajectories (Figure 3.9C, top right).

We next performed an analysis qualitatively similar to Li et al.'s. We identified a "coding subspace" (CS) that accounts for most of the across-condition variance in firing rates towards the end of movement preparation in unperturbed trials (Section 3.4.6). Similarly, we identified a "persistent" subspace (PS) that captures most of the activity difference between perturbed and unperturbed trials towards the end of preparation, regardless of the reach direction. Finally, we also contructed a third subspace, constrained to be orthogonal to the PS and the CS, and capturing most of the remaining variance across the different reaches and perturbation conditions

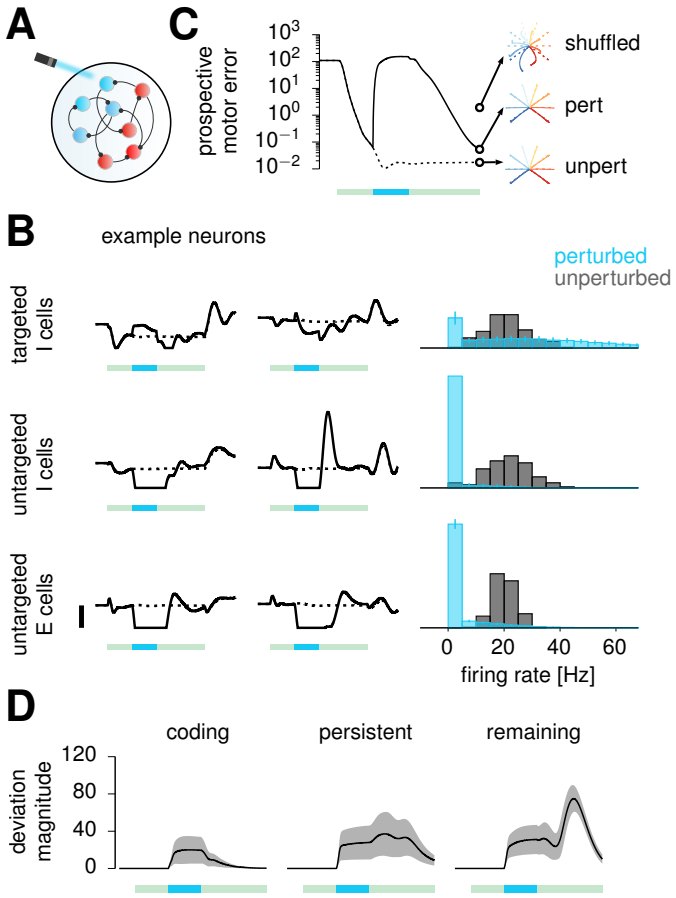


Figure 3.9: **Testable prediction: selective recovery from preparatory perturbations. (A)** Illustration of perturbation via "photoinhibition": a subset (60%) of I neurons in the model are driven by strong positive input. **(B)** Left: firing rates (solid: perturbed; dashed: unperturbed) for a pair of targeted I cells (top), untargeted I cells (middle) and E cells (bottom). Green bars (1.6 s) mark the movement preparation epoch, and embedded turquoise bars (400 ms) denote the perturbation period. Right: histogram of firing rates observed at the end of the perturbation (turquoise), and at the same time in unperturbed trials (gray). Error bars show one standard deviation across 300 experiments, each with a different random set of targeted I cells. **(C)** Prospective motor error, averaged across movements and perturbation experiments, in perturbed (solid) vs. unperturbed (dashed) conditions. Subsequent hand trajectories are shown for one experiment of each condition (middle and bottom insets; dashed lines show target trajectories). These are compared with the reaches obtained by randomly shuffling final preparatory errors $\delta \boldsymbol{x}$ across neurons, and re-simulating the cortical dynamics thereafter (top inset; see also text). **(D)** Magnitude of the deviation caused by the perturbation in the activity of the network projected into the coding subspace (left), the persistent subspace (center) and the remaining subspace (right). Lines denote the mean across perturbation experiments, and shadings indicate ± 1 s.t.d. Green and turquoise bars as in (B).

("remaining subspace", RS).

We found the CS and the PS to be nearly orthogonal (minimum subspace angle of 89 degrees) even though they were not constrained to be so. Moreover, the perturbation causes cortical activity to transiently deviate from unperturbed trajectories nearly equally in each of the three subspaces (Figure 3.9D). Remarkably, however, activity recovers promptly along the CS, but not along the other two modes. In fact, the perturbation even grows transiently along the PS during early recovery. Such selective recovery can be understood from optimal preparation eliminating errors along directions with motor consequences, but (owing to energy constraints) not in other inconsequential modes. Indeed, the CS is by definition a prospectively potent subspace: its contribution to the preparatory state is what determines the direction of the upcoming reach. In contrast, the PS and the RS are approximately prospectively null (respectively 729 times and 8 times less potent than the CS, by our measure $\mathcal{C}$ of motor potency in Figure 3.3D). This not only explains why the CS and PS are found to be near-orthogonal, but also why the thalamocortical dynamics implementing optimal preparatory control only quench perturbation-induced deviations in the CS.

In summary, optogenetic perturbations of M1 could be used to test a core prediction of optimal preparatory control, namely the selective recovery of activity in subspaces that carry movement information, but not in others.

3.3 Discussion

Neural population activity in cortex can be accurately described as arising from low-dimensional dynamics [8, 16, 21, 27, 110, 118, 163]. These dynamics unfold from a specific initial condition on each trial, and indeed these "preparatory states" predict the subsequent evolution of both neural activity and behaviour in single trials of the task [22, 131, 145, 169]. In addition, motor learning may rely on these preparatory states partitioning the space of subsequent movements [165].

How are appropriate initial conditions reached in the first place? Here, we have formalized movement preparation as an optimal control problem, showing how to translate anticipated motor costs phrased in terms of muscle kinematics into costs on neural activity in M1. Optimal preparation minimizes these costs, and the solution is feedback control: the cortical network must provide corrective feedback to itself, based on prospective motor errors associated with its current state. In other words, optimal preparation may rely on an implicit forward model predicting the future motor consequences of *preparatory activity*—not motor commands, as in

classical theories [31, 160, 207]—and feeding back these predictions for online correction of the cortical trajectory. Thus, where previous work has considered the motor cortex as a controller of the musculature [100, 178, 184], our work considers M1 and the body jointly as a compound system under preparatory control by other brain areas.

One of the key assumptions of this work is that the cortical dynamics responsible for reaching are at least *partially* determined by the inital state (we do not assume that they are autonomous). If, on the contrary, M1 activity were purely input-driven, there would be no need for preparatory control, let alone an optimal one. On the one hand, perturbation experiments in mice suggest thalamic inputs are necessary for continued movement generation and account for a sizeable fraction (though not all) of cortical activity during movement production [157]. On the other hand, experiments in non-human primates provide ample evidence that preparation does occur, irrespective of context [95], and that initial pre-movement states influence subsequent behaviour [1, 22, 166]. Thus, preparatory control is likely essential for accurate movement production.

3.3.1 Sloppy preparation for accurate movements

A core insight of our analysis is that preparatory activity must be constrained in a potent subspace impacting future motor output, but is otherwise free to fluctuate in a nullspace. This readily explains two distinctive features of preparatory activity in reaching monkeys: (i) that pre-movement activity on zero-delay trials needs not reach the state achieved for long movement delays [3], and (ii) that nevertheless movement is systematically preceded by activity in the same preparatory subspace irrespective of whether the reach is self-initiated, artificially delayed, or reactive and fast [95]. In our model, preparatory activity converges rapidly in the subspace that matters, such that irrespective of the delay (above 50 ms), preparatory activity is always found to have some component in this common subspace as in Lara et al. [95]. Moreover, exactly which of the many acceptable initial conditions is reached by the end of the delay depends on the delay duration. Thus, our model predicts that different preparatory end-states will be achieved for short and long delays, consistent with the results of Ames et al. [3]. Finally, these end-states also depends on the activity prior to preparation onset. This would explain why Ames et al. [3] observed different pre-movement activity states when preparation started from scratch and when it was initiated by a change in target that interrupted a previous preparatory process. In the same vein, Sauerbrei et al. [157] silenced thalamic input to mouse M1 early during movement, and observed that M1 activity did not recover to the pre-movement activity seen in unperturbed trials, even when the movement was successfully performed following the perturbation. This

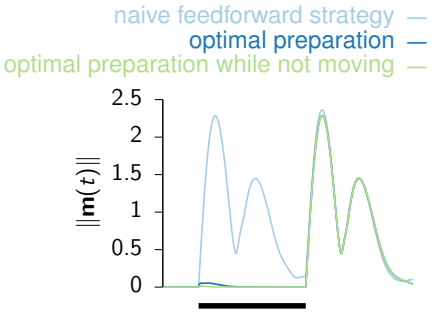


Figure 3.10: **Preparing whilst not moving.** Norm of the output joint torques, $\|\boldsymbol{m}(t)\|$, averaged over the 8 reaches, for the naive feedforward strategy (light blue), optimal preparatory control (dark blue), and an extended optimal preparatory control strategy that explicitly penalizes premature movement (green). The black bar marks the preparatory epoch.

result is expected in our model if the perturbation is followed by a new preparatory phase in which the effect of the perturbation rapidly vanishes along prospectively potent dimensions, but subsists in the nullspace.

3.3.2 Preparing without moving

How can preparatory M1 activity not cause premature movement, if M1 directly drives muscles? Kaufman et al. [87] argued that cortical activity may evolve in a coordinated way at the population level so as to remain in the nullspace of the muscle readout. Here, we have not explicitly penalized premature movements as part of our control objective (Equation (3.5)). As a result, while preparatory activity enters the nullspace eventually —because we constrained the movement-seeding initial conditions to belong to it—, optimal preparatory control yields small activity excursions outside the nullspace early during preparation (Figure 3.10), causing the hand to drift absent any further gating mechanism. Nevertheless, one can readily penalize such premature movements in our framework and obtain network models that prepare rapidly "in the nullspace" via closed-loop feedback (Figure 3.10; Section 3.4.5). Importantly, a naive feedforward control strategy is unable to contain the growth of movement-inducing activity during preparation.

3.3.3 Thalamic control of cortical dynamics

The mathematical structure of the optimal control solution suggested a circuit model based on cortico-cortical feedback. We have proposed that optimal feedback can be implemented as a cortico-thalamo-cortical loop, switched ON during movement preparation and OFF again at

movement onset. The ON-switch occurs through fast disinhibition of those thalamic neurons that are part of the loop. Our model thus predicts a large degree of specificity in the synaptic interactions between cortex and thalamus [67, 76], as well as a causal involvement of the thalamus in movement preparation [63, 157]. Furthermore, the dynamical entrainment of thalamus with cortex predicts tuning of thalamic neurons to task variables, consistent with a growing body of work showing specificity in thalamic responses [63, 123, 149]. For example, we predict that neurons in the motor thalamus should be tuned to movement properties, for much the same reasons that cortical neurons are [100, 127, 184]. Finally, we speculate that an ON/OFF switch on the thalamocortical loop is provided by one of the output nuclei of the basal ganglia (SNR or GPi), or a subset of neurons therein. Thus, exciting these midbrain neurons during preparation should prevent the thalamus from providing the necessary feedback to the motor cortex, thereby slowing down the preparatory process and presumably increasing reaction times. In contrast, silencing these neurons during movement should close the preparatory thalamocortical loop at a time when thalamus would normally disengage. This should modify effective connectivity in the movement-generating cortical network and impair the ongoing movement. These are predictions that could be tested in future experiments.

Thalamic control of cortical dynamics offers an attractive way of performing nonlinear computations [103, 175]. Although both preparatory and movement-related dynamics are approximately linear in our model, the transition from one to the other (orchestrated by the basal ganglia) is highly nonlinear. Indeed, our model can be thought of as a switching linear dynamical system [101]. Moreover, gated thalamocortical loops are a special example of achieving nonlinear effects through gain modulation. Here, it is the thalamic population only that is subjected to abrupt and binary gain modulation, but changes in gain could also affect cortical neurons. This was proposed recently as a way of expanding the dynamical repertoire of a cortical network [173].

Switch-like nonlinearities may have relevance beyond movement preparation, e.g. for movement execution. In our model, different movement patterns are produced by different initial conditions seeding the same generator dynamics. However, we could equally well have generated each reach using a different movement-epoch thalamocortical loop. Logiaco et al. have recently explored this possibility, showing that gated thalamocortical loops provide an ideal substrate for flexible sequencing of multiple movements. In their model, each movement is achieved by its own loop (involving a shared cortical network), and the basal ganglia orchestrate a chain of thalamic disinhibitory events, each spatially targeted to activate those neurons that are responsible for the next loop in the sequence [103]. Interestingly, their cortical network must still be properly initialized prior to each movement chunk, as it must in our model. For this, they proposed a

generic preparatory loop similar to the one we have studied here. However, theirs does not take into account the degeneracies in preparatory states induced by prospective motor costs, which ours exploits. In sum, our model and theirs address complementary facets of motor control (preparation and sequencing), and could be combined into a single model.

To conclude, we have proposed a new theory of movement preparation and studied its implications for various models of movement-generating dynamics in M1. While these models capture several salient features of movement-epoch activity, they could be replaced by more accurate, data-driven models [131] in future work. This would enable our theory to make detailed quantitative predictions of preparatory activity, which could be tested further in combination with targeted perturbation experiments. Our control-theoretic framework could help elucidate the role of the numerous brain areas that collectively control movement [179], and make sense of their hierarchical organization in nested loops.

3.4 Methods

3.4.1 A model for movement generation by cortical dynamics

Parameters

The values of all the parameters used in this study are listed in Table 3.1.

Network dynamics

We model M1 as a network with two separate populations of $N_\mathrm{E} = 160$ excitatory (E) neurons and $N_\mathrm{I} = 40$ inhibitory (I) neurons, operating in the inhibition-stabilized regime [73, 128, 190]. We constructed its synaptic architecture using the algorithm we have previously described in Hennequin et al. [73]. Briefly, we iteratively updated the inhibitory synapses of a random network, with an initial spectral abscissa of 1.2, to minimize a measure of robust network stability. We implemented early stopping, terminating the stabilization procedure as soon as the spectral abscissa of the connectivity matrix dropped below $\sim$0.8.

We describe the dynamics of these $N = N_\mathrm{E} + N_\mathrm{I}$ neurons by a standard nonlinear rate equation. Specifically, the vector $\boldsymbol{x}(t) = \left(\boldsymbol{x}_\mathrm{E}(t)^T, \boldsymbol{x}_\mathrm{I}(t)^T\right)^T$ of internal neuronal "activations" obeys:

$$\tau \frac{d\boldsymbol{x}}{dt} = -\boldsymbol{x}(t) + \boldsymbol{W}\phi\left[\boldsymbol{x}(t)\right] + \overline{\boldsymbol{h}} + \boldsymbol{h}(t) + \boldsymbol{u}(t) \tag{3.7}$$

where τ is the single-neuron time constant, $\boldsymbol{W}$ is the synaptic connectivity matrix, and $\phi(x) =$

symbol	value	unit	description
			PARAMETERS OF THE M1 CIRCUIT MODEL
N_{E}	160	-	number of E units
N_{I}	40	-	number of I units
τ	150	ms	time constant of M1 dynamics
τ_{rise}	50	ms	rise time constant of $\boldsymbol{h}(t)$
τ_{decay}	500	ms	decay time constant of $\boldsymbol{h}(t)$
A	implicit	-	set so that $\boldsymbol{h}(t)$ has a maximum of 5
			ARM MECHANICS AND HAND TRAJECTORIES
L_1	30	cm	length of upper arm link
L_2	33	cm	length of lower arm link
M_1	1.4	kg	mass of upper arm link
M_2	1.0	kg	mass of lower arm link
D_2	16	cm	center of mass of lower link, away from elbow
I_1	0.025	kg m^{-2}	moment of inertia of upper link
I_2	0.045	kg m^{-2}	moment of inertia of lower link
θ_1^{init}	10.	deg.	value of θ_1 at rest
θ_2^{init}	143.54	deg.	value of θ_2 at rest
$\theta_{\mathrm{reach}}^{(i)}$	$36 \times (i-2)$	deg.	reach angles ($i = 1, \ldots, 8$)
d_{reach}	20	cm	reach distance
τ_{reach}	120	ms	time constant of reach velocity profile
			LQR SOLUTION
λ	0.1	-	input energy penalty in Equation (3.28)
			THALAMO-CORTICAL CIRCUIT MODEL
λ	0.01	-	input energy penalty in Equation (3.28)
p	0.2	-	density of random connections from M1 to thal.
M_{E}	100	-	number of E units in M1 L4
M_{I}	100	-	number of I units in M1 L4
τ_y	10	ms	neuronal time constant in thalamus
τ_z	10	ms	neuronal time constant in M1 L4
			PHOTOINHIBITION
N_{ph}	100	-	number of M1 I units perturbed (60%)
h_{ph}	3	-	input to perturbed I units during photoinhibition
T_{ph}	400	ms	duration of photoinhibition

Table 3.1: Generic parameters used throughout all simulations

$\max(x, 0)$ is a static, rectified-linear nonlinearity – applied elementwise to $\boldsymbol{x}$ – that converts internal activation into momentary firing rates. The input consists of three terms: an input $\overline{\boldsymbol{h}} = \boldsymbol{x}_{\mathrm{sp}} - \boldsymbol{W}\phi[\boldsymbol{x}_{\mathrm{sp}}]$ held constant throughout all phases of the task to instate a heterogeneous set of spontaneous firing rates $\boldsymbol{x}_{\mathrm{sp}}$ (elements drawn i.i.d. from $\mathcal{N}(20, 9)$); a transient, movement-

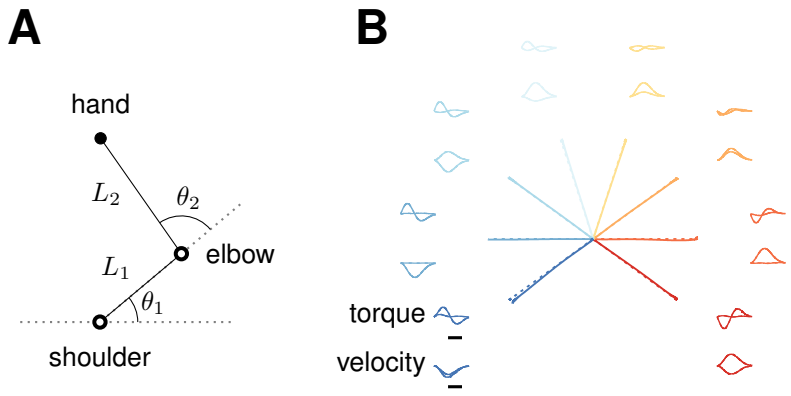


Figure 3.11: **(A)** Schematics of the arm model. **(B)** Reaches produced by the model, along with associated torques at the two joints, and x-y velocities of the hand (solid lines). Dashed lines denote target trajectories. Scale bar: 200 ms.

condition-independent and spatially uniform α-shaped input bump

$$\boldsymbol{h}(t) = (1, \ldots, 1)^T \times \begin{cases} \text{if } t > t_{\text{move}}: & A\left[\exp\left(-\frac{t-t_{\text{move}}}{\tau_{\text{decay}}}\right) - \exp\left(-\frac{t-t_{\text{move}}}{\tau_{\text{rise}}}\right)\right] \\ \text{otherwise:} & 0 \end{cases} \tag{3.8}$$

kicking in at movement onset [88]; and a preparatory control input $\boldsymbol{u}(t)$ (further specified below) whose role is to drive the circuit into a preparatory state appropriate for each movement.

We assume that the uncontrolled dynamics ($\boldsymbol{u} = \boldsymbol{0}$) of this network directly drives movement. To actuate the two-link arm model described in the next section, the activity of the network is read out into two joint torques:

$$\boldsymbol{m}(t) = \boldsymbol{C}\phi\left[\boldsymbol{x}(t)\right] \tag{3.9}$$

where $\boldsymbol{C} \in \mathbb{R}^{2\times N}$ is such that its last N_{I} columns are zero, i.e. only the excitatory neurons contribute directly to the motor output. Although our simulations show that the muscle readouts $\boldsymbol{m}(t)$ are very small during preparation, they do cause drift in the hand prior to movement onset (and therefore wrong movements afterwards) as they are effectively integrated twice by the dynamics of the arm (see below and Figure 3.10). For this reason, we artificially set $\boldsymbol{m}$ to zero during movement preparation.

Arm model

To simulate reaching movements, we used the planar two-link arm model previously described in Li and Todorov [99], with parameters listed in Table 3.1. The upper arm and the lower arm are connected at the elbow (Figure 3.11). The two links have lengths L_1 and L_2, masses M_1 and M_2, and moments of inertia I_1 and I_2 respectively. The lower arm's center of mass is located at

a distance D_2 from the elbow. By considering the geometry of the upper and lower limb, we can write down the position of the hand as a vector $\mathbf{y}(t)$ given by

$$\mathbf{y} = \begin{pmatrix} L_1 \cos\theta_1 + L_2 \cos(\theta_1 + \theta_2) \\ L_1 \sin\theta_1 + L_2 \sin(\theta_1 + \theta_2) \end{pmatrix} \tag{3.10}$$

where the angles θ_1 and θ_2 are defined in Figure 3.11A. The joint angles $\boldsymbol{\theta} = (\theta_1; \theta_2)^T$ evolve dynamically according to the differential equation

$$\mathcal{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathcal{X}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + \mathcal{B}\dot{\boldsymbol{\theta}} = \boldsymbol{m}(t), \tag{3.11}$$

where $\boldsymbol{m}(t)$ is the momentary torque vector (the output of the neural network, c.f. Equation (3.9)), $\mathcal{M}$ is the matrix of inertia, $\mathcal{X}$ accounts for the centripetal and Coriolis forces, and $\mathcal{B}$ is a damping matrix representing joint friction. These parameters are given by

$$\mathcal{M}(\boldsymbol{\theta}) = \begin{pmatrix} a_1 + 2a_2 \cos\theta_2 & a_3 + a_2 \cos\theta_2 \\ a_3 + a_2 \cos\theta_2 & a_3 \end{pmatrix} \tag{3.12}$$

$$\mathcal{X}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = a_2 \sin\theta_2 \begin{pmatrix} -\dot{\theta}_2(2\dot{\theta}_1 + \dot{\theta}_2) \\ \dot{\theta}_1{}^2 \end{pmatrix} \qquad \mathcal{B} = \begin{pmatrix} 0.05 & 0.025 \\ 0.025 & 0.05 \end{pmatrix} \tag{3.13}$$

with $a_1 = I_1 + I_2 + M_2 L_1^2$, $a_2 = M_2 L_1 D_2$, and $a_3 = I_2$.

Target hand trajectories

We generated a set of eight target hand trajectories, namely straight reaches of size $d = 20$ cm going from the origin ($(0, d)$ from the shoulder) into eight different directions, with a common bell-shaped scalar speed profile

$$v(t) = v_0 \left(\frac{t}{\tau_{\text{reach}}}\right)^2 \exp\left[-\frac{1}{2}\left(\frac{t}{\tau_{\text{reach}}}\right)^2\right], \tag{3.14}$$

where v_0 is adjusted such that the hand reaches the target. Given these target hand trajectories, we solved for the required timecourse of the torque vector $\boldsymbol{m}(t)$ through optimization, by backpropagating through the equations of motion of the arm (discretized using Euler's method) to minimize the squared difference between actual and desired hand trajectories. We forced the initial torques at $t = 0$ to be zero, and also included a roughness penalty in the form of average squared torque gradient.

Similarly, we then backpropagated through the equations of the recurrent neural network (Equations (3.7) and (3.9)) to optimize a set of eight movement-specific initial conditions $\{\boldsymbol{x}_k^\star\}$, $k = 1, \ldots, 8$, as well as the readout matrix $\boldsymbol{C}$, so as to achieve the desired torques in the

output. This was done by minimizing the squared difference between actual and desired torque trajectories, with a penalty on $\boldsymbol{C}$'s squared Frobenius norm.

We parameterized the readout matrix $\boldsymbol{C}$ in such a way that its nullspace automatically contains both the spontaneous activity vector $\boldsymbol{x}_{\text{sp}}$ and the movement-specific initial conditions $\{\boldsymbol{x}_k^\star\}$, $k = 1, \ldots, 8$. This is to ensure that (i) there is no muscle output during spontaneous activity and (ii) the network does not unduly generate muscle output at the end of preparation, before movement. More specifically, prior to movement preparation and long enough after movement execution, the cortical state is in spontaneous activity $\boldsymbol{x}_{\text{sp}}$. By ensuring that $\boldsymbol{C}\phi(\boldsymbol{x}_{\text{sp}}) = \boldsymbol{C}\boldsymbol{x}_{\text{sp}} = \boldsymbol{0}$, we ensure that our model network does not elicit movement "spontaneously". Similarly, control inputs drive the cortical state $\boldsymbol{x}$ towards $\boldsymbol{x}_k^\star$, which it will eventually reach late in the preparation epoch; therefore, if we did not constrain $\boldsymbol{x}_k^\star$ to be in $\boldsymbol{C}$'s null-space, premature movements would be elicited towards the end of preparation.

Training of other network classes

In Figures 3.5 and 3.7, we considered three other network types: 'full', 'low-rank', and 'chaotic' networks, which differed from the ISN in the way we constructed their synaptic connectivity matrices. We implemented 10 independent instances of each class.

Synaptic weights in the chaotic networks were drawn from a normal distribution with mean $-25/N$ and variance $1.8^2/N$ [83]. The connectivity matrices of the full and low-rank networks were obtained through optimization. Specifically, we jointly optimized $\boldsymbol{W}$, $\{\boldsymbol{x}_k^\star\}_{k=1,\ldots,8}$, and $\boldsymbol{C}$ to minimize errors in output torque just as described above for the ISN. Following Sussillo et al. [178], we also added a penalty on the squared Frobenius norm of both $\boldsymbol{C}$ and $\boldsymbol{W}$. While we optimized every element of $\boldsymbol{W}$ for the full networks, we parameterized the low-rank networks as

$$\boldsymbol{W} = \boldsymbol{W}_{\text{base}} + \boldsymbol{u}\boldsymbol{v}^T, \tag{3.15}$$

where $\boldsymbol{u}\boldsymbol{v}^T$ is a rank-5 perturbation to a random connectivity matrix $\boldsymbol{W}_{\text{base}}$. Here, both $\boldsymbol{u}$ and $\boldsymbol{v} \in \mathbb{R}^{N\times 5}$ are free parameters, whereas $\boldsymbol{W}_{\text{base}}$ is a fixed matrix with elements drawn anew for each network instance from $\mathcal{N}(0, 0.9^2/N)$.

We found that the addition of the reach-independent input $\boldsymbol{h}(t)$ during the movement epoch (Equation (3.2)) made these networks difficult to train: they either became unstable, entered a chaotic regime, or were unable to produce the desired movements altogether. Thus, we excluded $\boldsymbol{h}(t)$ for all these networks, as well as for the 10 other ISNs that we constructed as part of this multi-network comparison.

Similarity transformation of network solutions

A trained (linear) network's solution is completely determined by the triplet $\mathcal{S} = (\boldsymbol{A}, \boldsymbol{C}, \{\boldsymbol{x}_k^\star\})$, where $\boldsymbol{A} \triangleq \boldsymbol{I} - \boldsymbol{W}$ is the state transition matrix, $\boldsymbol{C}$ is the linear readout, and $\{\boldsymbol{x}_k^\star\}$ for $k = 1, \cdots, 8$ is the set of movement-specific initial conditions. Using an invertible transformation $\mathcal{T}$, we can construct a new solution given by the triplet $\tilde{\mathcal{S}} = (\tilde{\boldsymbol{A}}, \tilde{\boldsymbol{C}}, \{\tilde{\boldsymbol{x}}_k^\star\})$, where

$$\tilde{\boldsymbol{A}} = \mathcal{T}^{-1}\boldsymbol{A}\mathcal{T}, \quad \tilde{\boldsymbol{C}} = \boldsymbol{C}\mathcal{T}, \quad \text{and} \quad \tilde{\boldsymbol{x}}_k^\star = \boldsymbol{x}_{\text{sp}} + \mathcal{T}^{-1}(\boldsymbol{x}_k^\star - \boldsymbol{x}_{\text{sp}}), \tag{3.16}$$

and $\overline{\boldsymbol{h}}$ changed to $-\tilde{\boldsymbol{A}}\boldsymbol{x}_{\text{sp}}$ to ensure the spontaneous state remains the same. To see that $\tilde{\mathcal{S}}$ is also a solution to the task, we note that the output torques $\boldsymbol{m}_k^\star(t)$ at any time t are equal to

$$\boldsymbol{m}_k^\star(t) = \boldsymbol{C}e^{(t/\tau)\boldsymbol{A}}\boldsymbol{x}_k^\star = \tilde{\boldsymbol{C}}e^{(t/\tau)\tilde{\boldsymbol{A}}}\tilde{\boldsymbol{x}}_k^\star = \tilde{\boldsymbol{m}}_k^\star(t). \tag{3.17}$$

In Figure 3.7, we build the surrogate networks by applying the similarity transformation $\mathcal{T}$ that (block-)diagonalizes $\boldsymbol{A}$, resulting in $\tilde{\boldsymbol{A}}$ of the form:

$$\tilde{\boldsymbol{A}} = \begin{pmatrix} \ddots & & & & & \\ & c_j & & & & \\ & & \ddots & & & \\ & & & a_k & b_k & \\ & & & -b_k & a_k & \\ & & & & & \ddots \end{pmatrix} \tag{3.18}$$

where $\{c_k\}$ and $\{a_k \pm ib_k\}$ are the sets of real and complex-conjugate eigenvalues of $\boldsymbol{A}$. The resulting $\tilde{\boldsymbol{A}}$ is a "normal" matrix ($\tilde{\boldsymbol{A}}\tilde{\boldsymbol{A}}^T = \tilde{\boldsymbol{A}}^T\tilde{\boldsymbol{A}}$; 189), which—contrary to the original matrix $\boldsymbol{A}$—is unable to transiently amplify the set of movement-specific initial states $\{\tilde{\boldsymbol{x}}_k^\star\}$ during movement (Figure 3.7D). In order to be able to meaningfully compare $\|\boldsymbol{C}\|_{\text{F}}$ and $\|\tilde{\boldsymbol{C}}\|_{\text{F}}$ in Figure 3.7F, we scale $\mathcal{T}$ so as to preserve the average magnitude of the initial states (relative to the spontaneous state):

$$\sum_k \|\tilde{\boldsymbol{x}}_k^\star - \boldsymbol{x}_{\text{sp}}\|^2 = \sum_k \|\boldsymbol{x}_k^\star - \boldsymbol{x}_{\text{sp}}\|^2. \tag{3.19}$$

We can always achieve this because $\tilde{\boldsymbol{A}} = \mathcal{T}^{-1}\boldsymbol{A}\mathcal{T} = (\alpha\mathcal{T})^{-1}\boldsymbol{A}(\alpha\mathcal{T})$ for any $\alpha \neq 0$.

3.4.2 Formalization of anticipatory motor control

We formalise the notion of anticipatory control by asking: given an intended movement (indexed by k), and the current (preparatory) state $\boldsymbol{x}(t)$ of the network, how accurate would the movement be if it were to begin *now*? We measure this prospective motor error as the total squared

difference $\mathcal{C}_k(\boldsymbol{x})$ between the timecourses of the target torques $\boldsymbol{m}^\star$ and those that the network would generate (Equations (3.7) and (3.9)) *if left uncontrolled* from time t onwards, starting from initial condition $\boldsymbol{x}(t)$:

$$\mathcal{C}_k(\boldsymbol{x}) \triangleq \int_t^\infty \frac{dt'}{\tau} \left\|\boldsymbol{m}(t') - \boldsymbol{m}_k^\star(t'-t)\right\|^2 \quad \text{with} \quad \boldsymbol{u}(t' \geq t) = 0 \tag{3.20}$$

(we will often drop the explicit reference to the movement index k to remove clutter, as we did in the main text). Thus, any preparatory state $\boldsymbol{x}$ is associated with a prospective motor error $\mathcal{C}(\boldsymbol{x})$.

The prospective error $\mathcal{C}(\boldsymbol{x})$ changes dynamically during movement preparation, as $\boldsymbol{x}(t)$ evolves under the action of control inputs. The aim of the control inputs is to rapidly decrease this prospective error, until it drops below an acceptably small threshold, or until movement initiation is forced. We formalize this as the minimization of the following control cost:

$$\mathcal{J}\left[\boldsymbol{u}(t)\right] \triangleq \left\langle \int_0^\infty \frac{dt}{\tau} \left(\mathcal{C}\left(\boldsymbol{x}(t)\right) + \lambda \mathcal{R}\left(\boldsymbol{u}(t)\right)\right) \right\rangle_{p(\boldsymbol{x}(t=0))} \tag{3.21}$$

where $\mathcal{R}(\boldsymbol{u})$ is a regularizer (see below) and the average $\langle\cdot\rangle$ is over some distribution of states we expect the network to be found in at the time the controlled preparatory phase begins (we leave this unspecified for now as it turns out not to influence the optimal control strategy – see below). Thus, we want control inputs to rapidly steer the cortical network into states of low $\mathcal{C}(\boldsymbol{x})$ from which the movement can be readily executed, but these inputs should not be too large. The infinite-horizon summation expresses uncertainty about how long movement preparation will last, and indeed encourages the network to be "ready" as soon as possible.

Mathematically, $\mathcal{J}[\cdot]$ is a functional of the spatio-temporal pattern of control input $\boldsymbol{u}(t)$ — indeed, $\boldsymbol{x}(t)$ depends on $\boldsymbol{u}(t)$ through Equation (3.7). The regularizer $\mathcal{R}(\boldsymbol{u})$, or "control effort", encourages small control inputs and will be specified later. Without regularization, the problem is ill-posed, as arbitrarily large control inputs could be used to instantaneously force the network into the right preparatory state in theory, leading to physically infeasible control solutions in practice. Also note that Equation (3.21) is an "infinite-horizon" cost, i.e. the integral runs from the beginning of movement preparation when control inputs kick in, until infinity. This does *not* mean, however, that the preparation phase must be infinitely long. In fact, good control inputs should (and will!) bring the integrand close to zero very fast, such that the movement is ready to begin after only a short preparatory phase (see e.g. Figure 3.4A in the main text).

In order to derive the optimal control law, we further assume that the dynamics of the network remain approximately linear during both movement preparation and execution. This is a good approximation provided only few neurons become silent in either phase (the saturation at zero firing rate is the only source of nonlinearity in our model, c.f. $\phi(\cdot)$ in Equation (3.7); Figure 3.12).

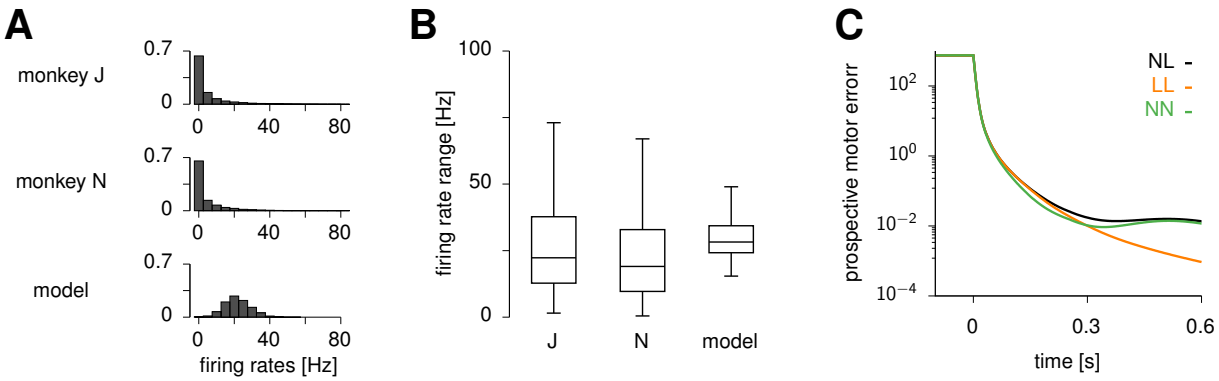


Figure 3.12: **Role of the nonlinearity in preparatory control. (A)** Distribution of firing rates across all neurons, times (including both preparation and movement epochs) and reach conditions. **(B)** Firing rate range (max − min) across time and reach conditions, computed for each neuron in monkey J, monkey N, and the model (min, 25th percentile, median, 75th percentile, max). **(C)** Prospective motor error during optimal preparation, under three different assumptions. NL: nonlinear network dynamics during preparation, but prospective error computed assuming linear dynamics during movement (c.f. Figure 3.4A of the main text). LL: same as NL, but with linear dynamics during preparation. NN: nonlinear dynamics during preparation, and prospective error computed using rollouts of the same nonlinear network dynamics during the movement-epoch. In all three cases, the optimal preparatory inputs were computed assuming a linear model as throughout the paper.

In this case, the prospective motor error $\mathcal{C}(\boldsymbol{x})$ of Equation (3.20) affords a simpler, interpretable form, which we derive now. In the linear regime, Equation (3.7) becomes

$$\tau \frac{d\boldsymbol{x}}{dt} = \boldsymbol{A}\boldsymbol{x}(t) + \overline{\boldsymbol{h}} + \boldsymbol{h}(t) + \boldsymbol{u}(t) \tag{3.22}$$

with an effective state transition matrix $\boldsymbol{A} \triangleq \boldsymbol{W} - \boldsymbol{I}$. The network output at time t', starting from state $\boldsymbol{x}$ at time $t' = 0$ and with no control input thereafter, has an analytical form given by

$$\boldsymbol{m}(t') = \boldsymbol{C}\left[e^{(t'/\tau)\boldsymbol{A}}(\boldsymbol{x} - \boldsymbol{x}_{\text{sp}}) + \boldsymbol{q}(t')\right] \tag{3.23}$$

and similarly for $\boldsymbol{m}^\star(t')$ with $\boldsymbol{x}$ replaced by $\boldsymbol{x}^\star$. The final term $\boldsymbol{q}(t')$ is a contribution from the external input: it does not depend on the initial condition, and is therefore the same in both cases. Thus, the prospective motor error (Equation (3.20)) attached to a given preparatory state $\boldsymbol{x}$ is

$$\mathcal{C}(\boldsymbol{x}) = \int_0^\infty \frac{dt'}{\tau} \left\|\boldsymbol{C}e^{(t'/\tau)\boldsymbol{A}}(\boldsymbol{x} - \boldsymbol{x}^\star)\right\|^2 = (\boldsymbol{x} - \boldsymbol{x}^\star)^T \underbrace{\left[\int_0^\infty \frac{dt'}{\tau}\left(e^{(t'/\tau)\boldsymbol{A}^T}\boldsymbol{C}^T\boldsymbol{C}e^{(t'/\tau)\boldsymbol{A}}\right)\right]}_{\boldsymbol{Q}}(\boldsymbol{x} - \boldsymbol{x}^\star). \tag{3.24}$$

The matrix integral on the r.h.s. of Equation (3.24) is known as the "observability Gramian" $\boldsymbol{Q}$ of the pair $(\boldsymbol{A}, \boldsymbol{C})$ [86, 168]. It is found algebraically as the solution to the Lyapunov equation[1]

$$\boldsymbol{A}^T\boldsymbol{Q} + \boldsymbol{Q}\boldsymbol{A} + \boldsymbol{C}^T\boldsymbol{C} = \boldsymbol{0}. \tag{3.25}$$

Thus, under linearity assumptions, the prospective motor error is a quadratic function of the difference between the momentary preparatory state $\boldsymbol{x}$ and an optimal initial state $\boldsymbol{x}^\star$ known to elicit the right muscle outputs in open loop. The Gramian $\boldsymbol{Q}$, a symmetric, positive-definite matrix, determines how preparatory deviations away from $\boldsymbol{x}^\star$ give rise to subsequent motor errors. Deviations along the few eigenmodes of $\boldsymbol{Q}$ associated with large eigenvalues will lead to large errors in muscle outputs. The optimal control input $\boldsymbol{u}(t)$ will need to work hard to minimize this type of 'prospectively potent' deviations – luckily, there are few (see Figure 3.3D in the main text, where the top 20 eigenvalues of $\boldsymbol{Q}$ are shown; see also Section 3.4.3 for further dissection). In contrast, errors occurring along eigenmodes of $\boldsymbol{Q}$ with small eigenvalues – the vast majority – have almost no motor consequences ('prospectively null'). This large bottom subspace of $\boldsymbol{Q}$ provides a safe buffer in which preparatory activity is allowed to fluctuate without sacrificing control quality. It comprises both the "readout-null" and "dynamic-null" directions described in the main text (Figure 3.3). Geometrically, we can therefore think of the optimal preparatory subspace as a high-dimensional ellipsoid centered on $\boldsymbol{x}^\star$, and whose small and (potentially inifinitely) large axes are given by the top and bottom eigenvectors of $\boldsymbol{Q}$, respectively (small axes, steep directions, large eigenvalues; long axes, flat directions, small eigenvalues).

Finally, we note that the optimal control input $\boldsymbol{u}(t)$ must keep the infinite-horizon integral in Equation (3.21) finite. This is achieved if $\boldsymbol{x}(t)$ reaches a fixed point equal to $\boldsymbol{x}^\star$, which is in turn achieved if the control input eventually settles in a steady state equal to

$$\boldsymbol{u}^\star = -\boldsymbol{A}\boldsymbol{x}^\star - \overline{\boldsymbol{h}} \tag{3.26}$$

Thus, defining $\delta\boldsymbol{u}(t) \triangleq \boldsymbol{u}(t) - \boldsymbol{u}^\star$ and $\delta\boldsymbol{x}(t) \triangleq \boldsymbol{x}(t) - \boldsymbol{x}^\star$, a relevant regularizer for our control problem is

$$\mathcal{R}\left(\boldsymbol{u}(t)\right) \triangleq \|\delta\boldsymbol{u}(t)\|^2 \tag{3.27}$$

and our control cost functional becomes

$$\mathcal{J}\left[\boldsymbol{u}(t)\right] = \left\langle \int_0^\infty \frac{dt}{\tau} \left[\delta\boldsymbol{x}(t)^T\boldsymbol{Q}\delta\boldsymbol{x}(t) + \lambda\|\delta\boldsymbol{u}(t)\|^2\right] \right\rangle_{p(\boldsymbol{x}(t=0))}. \tag{3.28}$$

[1] This result is central to the theory of linear quadratic control, where cost functions are often of the form of integrated squared functions of the state, output, or input, under linear dynamics. It allows one to manipulate these integrals algebraically, and compute them numerically by solving a linear matrix equation (e.g. 9). Indeed we will use this result again several times below in different contexts.

In our simulations, we perform a simple scalar normalization of $\boldsymbol{Q}$ so that $\mathrm{Tr}\,(\boldsymbol{Q}) = N$. This makes the first term of the cost more easily comparable to the energy penalty $\lambda\|\delta\boldsymbol{u}\|^2$, which also scales with N. In the next section, we show that the quadratic formulation of $\mathcal{J}$ in Equation (3.28) leads to analytically tractable optimization. In the rest of the Star Methods, as in the main text, we continue to assume linear network dynamics in order to derive optimal control laws but implement these solutions in the fully nonlinear circuit.

Classical LQR solution

When no specific constraints on $\boldsymbol{u}(t)$ are imposed, the minimization of Equation (3.28) is given by the celebrated linear quadratic regulator (LQR). Specifically, the optimal control input $\boldsymbol{u}_{\mathrm{opt}}(t) = \boldsymbol{u}^\star + \delta\boldsymbol{u}_{\mathrm{opt}}(t)$ takes the form of instantaneous linear state feedback (Figure 3.14A):

$$\delta\boldsymbol{u}^{\mathrm{opt}}(t) = \boldsymbol{K}\,\delta\boldsymbol{x}(t) \qquad \text{with} \qquad \boldsymbol{K} = -\lambda^{-1}\boldsymbol{P} \tag{3.29}$$

where $\boldsymbol{P}$ is a symmetric, positive-definite matrix, obtained as the solution to the following Riccati equation:

$$\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \lambda^{-1}\boldsymbol{P}\boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0}. \tag{3.30}$$

We will recover this optimal feedback law in Section 3.4.6 as part of a more general mathematical derivation; for now, we refer to standard texts, e.g. 168.

Thus, to achieve optimal anticipatory control of fast movements, the best strategy for the preparatory phase is to feed back into the circuit a linearly weighted version of the momentary error signal $\delta\boldsymbol{x}(t)$. The optimal feedback matrix $\boldsymbol{K}$ turns out to not depend on the choice of distribution $p(\boldsymbol{x}(t=0))$. For a linear model, this also implies that $\boldsymbol{K}$ does not depend on the specific movement to be performed, i.e. on the specific state $\boldsymbol{x}^\star$ to be approached during preparation. Only the steady-state control input $\boldsymbol{u}^\star$ in Equation (3.26) is movement-specific.

The total control cost is given by $\mathcal{J} = \delta\boldsymbol{x}_0^T\boldsymbol{P}_\lambda\delta\boldsymbol{x}_0$, where $\delta\boldsymbol{x}_0$ is the deviation of the network state from $\boldsymbol{x}^\star$ at the onset of movement preparation (i.e., the state of the network at preparation onset is $\boldsymbol{x}^\star + \delta\boldsymbol{x}_0$). The corresponding total energy cost $\int\mathcal{R}dt$ is given by $\delta\boldsymbol{x}_0^T\boldsymbol{Y}\delta\boldsymbol{x}_0$ where $\boldsymbol{Y}$ is the solution to

$$\boldsymbol{A}_{\mathrm{cl}}^T\boldsymbol{Y} + \boldsymbol{Y}\boldsymbol{A}_{\mathrm{cl}} + \lambda^{-2}\boldsymbol{P}\boldsymbol{P} = 0, \tag{3.31}$$

and

$$\boldsymbol{A}_{\mathrm{cl}} \triangleq \boldsymbol{A} + \boldsymbol{K} = \boldsymbol{A} - \lambda^{-1}\boldsymbol{P} \tag{3.32}$$

is the effective state matrix governing the dynamics of the closed control loop. The associated integrated prospective motor cost is then given by $\int\mathcal{C}(\boldsymbol{x}(t))dt = \delta\boldsymbol{x}_0^T(\boldsymbol{P} - \lambda\boldsymbol{Y})\,\delta\boldsymbol{x}_0$. This is how

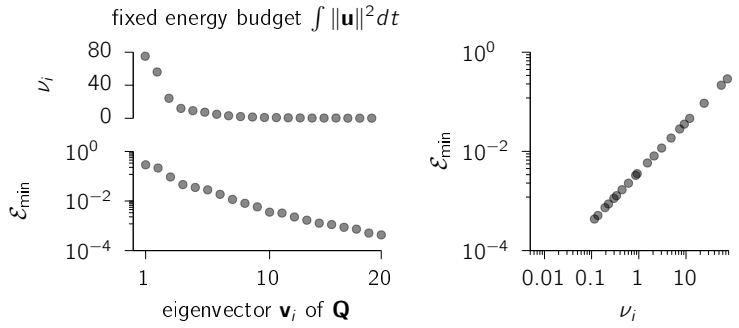


Figure 3.13: **Left:** eigenvalues ν_i (top) and minimum control cost $\mathcal{E}_{\min}$ achievable given a fixed energy budget (see text), for the top 20 eigenvectors of the observability Gramian $\boldsymbol{Q}$ defined in Equation (3.24). Note that ν_i is also the motor error $\mathcal{C}$ incurred by a deviation of the preparatory state $\boldsymbol{x}^\star$ of unit length in the direction of the corresponding eigenvector $\boldsymbol{v}_i$, just prior to movement. **Right:** same ν_i and $\mathcal{E}_{\min}$ as shown on the left, plotted against each other.

we evaluate the total energy cost and integrated prospective motor cost in Figure 3.5H.

We emphasize that the LQR problem (and its solution in Equation (3.29)) assumes linear dynamics throughout the preparatory and movement epochs to compute (and optimize) the prospective motor cost $\mathcal{C}(\boldsymbol{x}(t))$. We verified that the quadratic form of Equation (3.24) provides a good approximation to the actual $\mathcal{C}(\boldsymbol{x}(t))$ resulting from nonlinear dynamics during the movement epoch (Figure 3.12C). Moreover, all our simulations use the nonlinear activation function $\phi(\cdot)$ in Equation (3.7) unless indicated otherwise.

Naive feedforward solution

A straightforward solution exists for ensuring that, after enough preparation time, $\boldsymbol{x}(t)$ converges exponentially to $\boldsymbol{x}^\star$ – thus *eventually* leading to the correct movement. This "naive" solution consists in setting $\boldsymbol{u}(t)$ to the constant vector $\boldsymbol{u}^\star$ in Equation (3.26) (thus $\delta\boldsymbol{u}(t) = 0$ throughout preparation). Note that for the full nonlinear model, $\boldsymbol{u}^\star = \boldsymbol{x}^\star - \boldsymbol{W}\phi\left[\boldsymbol{x}^\star\right] - \overline{\boldsymbol{h}}$. This constant input is provided during movement preparation and removed at the desired time of movement onset. This naive strategy does not rely on feedback, and so can be seen as a type of feedforward preparatory control. It also corresponds to the LQR solution in the limit of infinite energy penalty λ in Equation (3.28), and indeed the solution in this limit yields $\boldsymbol{K} = 0$, resulting in $\mathcal{R}(\boldsymbol{u}(t)) = 0$.

3.4.3 Control geometry of the ISN

The optimal LQR strategy can be used to expose the challenges associated with controlling the inhibition-stabilized model of M1 that we use here. Indeed, network activity may be more easily controlled (or "steered") along some directions than along others, and having analytical access to the optimal control inputs (Equations (3.29) and (3.30)) allows us to quantify this "control geometry". Specifically, we quantify control performance as $\mathcal{E} = \int_0^\infty (\delta \boldsymbol{x}^T \boldsymbol{Q} \delta \boldsymbol{x}) dt$, i.e. our original cost functional $\mathcal{J}$ in Equation (3.21) without the input energy penalty. We can then ask: what is the smallest such cost $\mathcal{E}_{\min}$ that can be achieved with a fixed input energy budget $\int_0^\infty \|\delta \boldsymbol{u}\|^2 dt$? We know that $\mathcal{E}_{\min}$ is achieved by the LQR solution $\delta \boldsymbol{u} = \lambda^{-1} \boldsymbol{P}_\lambda$ (we use the $\cdot_\lambda$ subscript to make the dependence of $\boldsymbol{P}$ on λ explicit). It can be shown that the input energy induced by this optimal feedback law is a decreasing function of λ. Thus, all we need to do is find the λ that gives us the desired value of $\int_0^\infty \|\delta \boldsymbol{u}\|^2 dt$, and evaluate $\mathcal{E}_{\min}$ for this particular λ. Importantly, the result will depend on the state of the cortical network at the beginning of the controlled preparatory phase, relative to the target $\boldsymbol{x}^\star$.

A simple derivation based on the LQR Ricatti equation shows that starting the control phase from some initial condition $\boldsymbol{x}^\star + \delta \boldsymbol{x}_0$ yields a total control cost equal to $\mathcal{J} = \delta \boldsymbol{x}_0^T \boldsymbol{P}_\lambda \delta \boldsymbol{x}_0$. Moreover, the corresponding energy cost is given by $\delta \boldsymbol{x}_0^T \boldsymbol{Y} \delta \boldsymbol{x}_0$ where $\boldsymbol{Y}$ is the solution to

$$\boldsymbol{A}_{\text{cl}}^T \boldsymbol{Y} + \boldsymbol{Y} \boldsymbol{A}_{\text{cl}} + \lambda^{-2} \boldsymbol{P}_\lambda \boldsymbol{P}_\lambda = 0, \tag{3.33}$$

and

$$\boldsymbol{A}_{\text{cl}} \triangleq \boldsymbol{A} + \boldsymbol{K} = \boldsymbol{A} - \lambda^{-1} \boldsymbol{P}_\lambda \tag{3.34}$$

is the effective state matrix governing the dynamics of the closed control loop. For a given $\delta \boldsymbol{x}_0$, we use a simple root-finding method (bisection with initial interval bracketting) to find the λ that achieves the set, desired energy cost (our fixed "energy budget"). For this λ, we then calculate the associated control cost $\mathcal{E} = \delta \boldsymbol{x}_0^T (\boldsymbol{P} - \lambda \boldsymbol{Y}) \, \delta \boldsymbol{x}_0$. This is plotted in Figure 3.13, for initial deviations of $\boldsymbol{x}$ from $\boldsymbol{x}^\star$ chosen to be the top 20 eigenvectors of $\boldsymbol{Q}$, ranked by their respective eigenvalues ν_i (Equation (3.24)).

Figure 3.13 (right) shows that there is "no free lunch": preparatory deviations from $\boldsymbol{x}^\star$ that induce the worst motor errors (the top eigenvectors of $\boldsymbol{Q}$, with the largest eigenvalues ν_i) are also those that are the most difficult to control, i.e. for which the minimal control cost $\mathcal{E}_{\min}$ will be largest for a fixed input energy budget.

From the point of view of dynamical systems, this result is rather intuitive. The optimal initial conditions $\{\boldsymbol{x}_k^\star\}$ (found via optimization to achieve the required torques; Section 3.4.1) are

positioned in state space where the flow induced by the recurrent connectivity is strong – strong enough to elicit rich transients that can be decoded into torques patterns that grow transiently before decaying. To steer M1 towards (and maintain it at) these states, the input $\delta\boldsymbol{u}(t)$ (and the steady input $\boldsymbol{u}^\star$) must work against the strong local flow of the recurrent dynamics. This requires large inputs. From a physiological standpoint, this is also intuitive. The optimal initial states $\{\boldsymbol{x}_k^\star\}$ are shown to be states in which the E/I balance is momentarily broken [73]. Much input energy must be spent to sustain an E/I imbalance in a network whose connectivity strives to maintain balance.

3.4.4 Adapting optimal control to chaotic networks

In Figure 3.5D, we adapt optimal preparatory control to a chaotic network and find that LQR stabilizes its otherwise unstable dynamics during movement preparation. Thus far, we have assumed that $\boldsymbol{A}$ is Hurwitz-stable in deriving the optimal control law, which allows us to evaluate the integral in Equation (3.24) analytically by solving Equation (3.25). This integral diverges for unstable $\boldsymbol{A}$, and thus cannot be evaluated for a chaotic network. In this work, we simply set $\boldsymbol{Q} = \boldsymbol{I}$ for the chaotic network, which likely overestimates the number of state-space directions that matter for preparatory control. More sophisticated methods could potentially be used for estimating the sensitivity of these nonlinear dynamics to errors in the initial condition. This would presumably lead to faster preparation for a set energy budget $\mathcal{R}(\boldsymbol{u}(t))$, but we leave this for future work.

3.4.5 Preparing in the nullspace

In Figure 3.10, we extend the optimal control model to explicitly penalize non-zero output torques during movement preparation. Specifically, we augment the cost functional in Equation (3.28) with a penalty on the integrated squared magnitude of $\boldsymbol{m}(t)$, which can be written as a quadratic form in $\delta\boldsymbol{x}$ much like the prospective motor cost in Equation (3.24):

$$\int_0^\infty \frac{dt}{\tau} \|\boldsymbol{m}(t)\|^2 = \int_0^\infty \frac{dt}{\tau} \delta\boldsymbol{x}^T \boldsymbol{C}^T \boldsymbol{C} \delta\boldsymbol{x} \tag{3.35}$$

where $\boldsymbol{m}(t) = \boldsymbol{C}\delta\boldsymbol{x}$ because we have constrained the initial conditions $\{\boldsymbol{x}_k^\star\}$ to be in the nullspace of $\boldsymbol{C}$. We thus write the combined cost functional as

$$\mathcal{J}\left[\boldsymbol{u}(t)\right] = \left\langle \int_0^\infty \frac{dt}{\tau} \left[\delta\boldsymbol{x}(t)^T \tilde{\boldsymbol{Q}} \delta\boldsymbol{x}(t) + \lambda \|\delta\boldsymbol{u}(t)\|^2\right] \right\rangle_{p(\boldsymbol{x}(t=0))} \tag{3.36}$$

with

$$\tilde{\boldsymbol{Q}} = \eta \boldsymbol{Q} + (1-\eta) N \frac{\boldsymbol{C}^T \boldsymbol{C}}{\mathrm{Tr}(\boldsymbol{C}^T \boldsymbol{C})}, \tag{3.37}$$

where $\eta \in [0, 1]$ is a scalar that weighs the relative importance of preparing fast and preparing while not moving ($\eta = 0.2$ in Figure 3.10).

3.4.6 Optimality under neural constraints

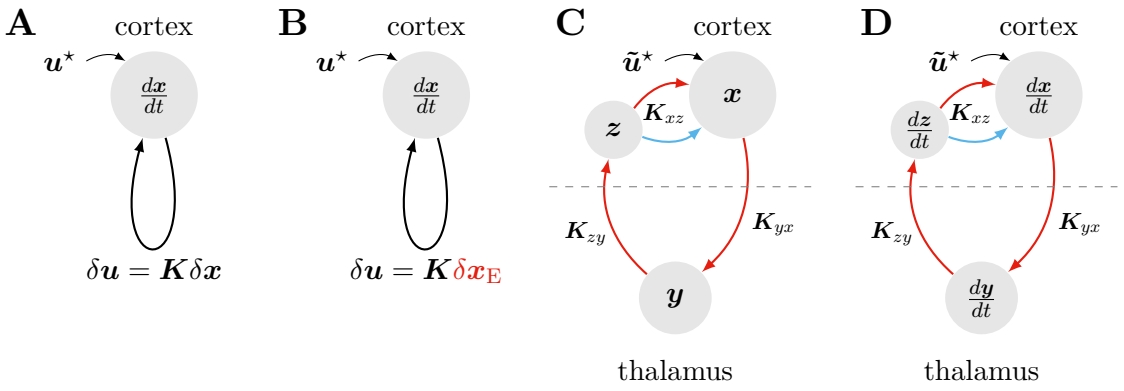


Figure 3.14: Four steps to arrive at a biologically plausible implementation of optimal anticipatory motor control. **(A)** The classical LQR solution prescribes instantaneous state feedback, with reentrant control inputs of the form of $\delta\boldsymbol{u}(t) = \boldsymbol{K}\delta\boldsymbol{x}(t)$ and a constant external input $\boldsymbol{u}^\star$. Section 3.4.2 shows how to obtain the optimal feedback matrix $\boldsymbol{K}$ in this case. **(B)** It is possible to constrain feedback to be of the form $\delta\boldsymbol{u}(t) = \boldsymbol{K}\,[\boldsymbol{I}_{N_\mathrm{E}}\ \boldsymbol{0}_{N_i}]\,\delta\boldsymbol{x}(t) = \boldsymbol{K}\delta\boldsymbol{x}_\mathrm{E}(t)$ instead. Section 3.4.6 shows how to obtain the optimal feedback matrix $\boldsymbol{K}$ in this case. **(C)** For flexibility, we propose that feedback be relayed by the motor thalamus, which is under the gating control of the basal ganglia. In Section 3.4.6, we show that the optimal feedback gain $\boldsymbol{K}$ obtained in (B) can be decomposed into sign-constrained matrices implementing E connections from M1 to thalamus ($\boldsymbol{K}_{yx}$), from thalamus to M1-layer 4 ($\boldsymbol{K}_{zy}$), and Dale-structured E/I connections from layer 4 back into the main recurrent M1 circuit ($\boldsymbol{K}_{xz}$). **(D)** Finally, first-order dynamics can be introduced in our model thalamus and M1-layer 4 neurons. We show in Section 3.4.6 how the lag introduced by such dynamics can be taken into account, to obtain a set of connections that achieve optimal anticipatory control of movement under these biological constraints.

The linear quadratic regulator presented in Section 3.4.2 brings the fundamental insight that control can (and in fact, should) be achieved via a feedback loop (Figure 3.14A). Such a loop could technically be embedded directly as a modification of the recurrent connectivity within M1, as all that matters for the control cost is the effective closed-loop state matrix $\boldsymbol{A} + \boldsymbol{K}$. However,

this would make it very difficult to switch the loop ON when movement preparation must begin, and OFF again when the movement is triggered. A more flexibly strategy would be to have the loop pass through another brain area, and gain-modulate this area (e.g. via inhibitory drive) to close or open the loop when appropriate.

A natural candidate structure for mediating such cortico-cortical feedback is the motor thalamus, which has been shown to be causally involved in movement preparation [63]. Importantly, basic anatomy and physiology pose constraints on the type of connectivity and dynamics around the control loop, such that we will have to adapt the classical LQR theory to derive plausible circuit mechanisms. In particular, the thalamus is not innervated by the local inhibitory interneurons of M1, so feedback will have to computed based on the activity of (some of) the excitatory cells only, precluding full-state feedback. Moreover, the optimal LQR gain matrix $\boldsymbol{K}$ (given by Equations (3.29) and (3.30)) contains both positive and negative elements with no structure; this violates Dale's law, i.e. that neurons can be either excitatory or inhibitory but are never of a mixed type. Finally, the classical LQR solution prescribes *instantaneous* state feedback, whereas thalamic neurons will have to integrate their inputs on finite timescales, thereby introducing some "inertia", or lag, in the feedback loop. In the rest of this section, we flesh out these biological constraints in more detail, and show that all of these limitations can be addressed mathematically, to eventually yield optimal control via a realistic thalamocortical feedback loop (see Figure 3.14B-D for a graphical overview).

Feedback based on excitatory neurons only

Here, we incorporate the key biological constraints that feedback from the cortex onto itself via the thalamus will have to originate from the excitatory cells only. Thus, instead of $\delta\boldsymbol{u}(t) = \boldsymbol{K}\delta\boldsymbol{x}(t)$, we look for a feedback matrix of the form (Figure 3.14B)

$$\boldsymbol{K} = \boldsymbol{Z}\boldsymbol{\Gamma} \tag{3.38}$$

where $\boldsymbol{\Gamma} \triangleq [\boldsymbol{I}_{N_\mathrm{E}}\ \boldsymbol{0}_{N_\mathrm{E}\times N_\mathrm{I}}]$ singles out the activity of the E neurons when computing the control input $\boldsymbol{K}\delta\boldsymbol{x}$, and $\boldsymbol{Z}$ is an $N \times N_\mathrm{E}$ matrix of free parameters. To gain generality (which we will need later), we also assume that the control input enters the network through a matrix $\boldsymbol{B}$, i.e. the closed-loop state matrix (Equation (3.32)) becomes $\boldsymbol{A}_\mathrm{cl} = \boldsymbol{A} + \boldsymbol{B}\boldsymbol{Z}\boldsymbol{\Gamma}$, and the energy penalty becomes $\lambda\|\boldsymbol{B}\delta\boldsymbol{u}(t)\|^2$. We now derive algebraic conditions of optimality for $\boldsymbol{Z}$, along with a gradient-based method to find the optimal $\boldsymbol{Z}$ that fulfills them.

First, we use the general result of Equation (3.25) (see also Footnote 1) to rewrite the cost

function $\mathcal{J}$ in Equation (3.28) as:

$$\mathcal{J}(\boldsymbol{Z}) = \mathrm{Tr}\,(\boldsymbol{P}) \tag{3.39}$$

where $\boldsymbol{P}$ satisfies

$$0 = \boldsymbol{G}(\boldsymbol{P}, \boldsymbol{Z}) \triangleq \boldsymbol{A}_{\mathrm{cl}}^T \boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}_{\mathrm{cl}} + \boldsymbol{Q} + \lambda \boldsymbol{\Gamma}^T \boldsymbol{Z}^T \boldsymbol{B}^T \boldsymbol{B} \boldsymbol{Z} \boldsymbol{\Gamma}. \tag{3.40}$$

Note that $\mathcal{J}$ in Equation (3.39) is now a function of the feedback matrix $\boldsymbol{K}$, and therefore of the parameter matrix $\boldsymbol{Z}$. To minimize $\mathcal{J}$ w.r.t. $\boldsymbol{Z}$ subject to the constraint in Equation (3.40), we introduce the Lagrangian:

$$\mathcal{L}(\boldsymbol{P}, \boldsymbol{Z}, \boldsymbol{S}) \triangleq \mathrm{Tr}\,(\boldsymbol{P}) + \mathrm{Tr}\,(\boldsymbol{G}(\boldsymbol{P}, \boldsymbol{Z})\, \boldsymbol{S})\,, \tag{3.41}$$

where $\boldsymbol{S}$ is a symmetric matrix of Lagrange multipliers (the matrix equality in Equation (3.40) is symmetric, thus effectively providing $N(N+1)/2$ constraints). After some matrix calculus, we obtain the following coupled optimality conditions:

$$0 = \partial\mathcal{L}/\partial\boldsymbol{P} = \boldsymbol{A}_{\mathrm{cl}}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{A}_{\mathrm{cl}}^T + \boldsymbol{I} \tag{3.42}$$

$$0 = \partial\mathcal{L}/\partial\boldsymbol{S} = \boldsymbol{G}(\boldsymbol{P}, \boldsymbol{Z}) \tag{3.43}$$

$$0 = \partial\mathcal{L}/\partial\boldsymbol{Z} = 2\boldsymbol{B}^T \left(\boldsymbol{P} + \lambda \boldsymbol{B}\boldsymbol{Z}\boldsymbol{\Gamma}\right) \boldsymbol{S}\boldsymbol{\Gamma}^T. \tag{3.44}$$

When the two Lyapunov equations Equations (3.42) and (3.43) are satisfied, the second term ($\mathrm{Tr}\,(\boldsymbol{G}\boldsymbol{S})$) in $\mathcal{L}$ vanishes, such that $\partial\mathcal{L}/\partial\boldsymbol{Z}$ of Equation (3.44) is in fact the gradient of $\mathrm{Tr}\,(\boldsymbol{P})$ w.r.t. $\boldsymbol{Z}$ subject to the algebraic constraint of Equation (3.40). We use this gradient equation, together with the L-BFGS optimizer [15] to find the optimal parameter matrix $\boldsymbol{Z}$. We then recover the optimal feedback gain matrix $\boldsymbol{K}$ according to Equation (3.38). We start each optimization by setting $\boldsymbol{Z} = \overline{\boldsymbol{K}}\boldsymbol{\Gamma}^T(\boldsymbol{\Gamma}\boldsymbol{\Gamma}^T)^{-1}$, where $\overline{\boldsymbol{K}}$ is the classical LQR solution to the same problem, such that $\boldsymbol{Z}$ is the least-square solution to $\overline{\boldsymbol{K}} = \boldsymbol{Z}\boldsymbol{\Gamma}$.

Dale's law

The previous subsection showed how to obtain a gain matrix $\boldsymbol{K}$ of size $N \times N_{\mathrm{E}}$ that implements optimal, instantaneous cortico-cortical feedback originating from the excitatory cells. However, this optimal matrix typically has a mix of positive and negative elements that are not specifically structured. To implement the more realistic feedback architecture shown in Figure 3.14C, implicating the motor thalamus and M1 layer 4 (M1-L4), we seek a decomposition of the form

$$\boldsymbol{K} \approx \underbrace{\boldsymbol{K}_{xz}}_{(+|-)}\,\underbrace{\boldsymbol{K}_{zy}}_{(+)}\,\underbrace{\boldsymbol{K}_{yx}}_{(+)} \tag{3.45}$$

where $\boldsymbol{K}_{yx}$ (M1 to thalamus) is an $N_\mathrm{E} \times N$ matrix of non-negative elements, $\boldsymbol{K}_{zy}$ (thalamus to M1-L4) is an $M \times N_\mathrm{E}$ matrix of non-negative elements, and $\boldsymbol{K}_{xz}$ (M1-L4 to the recurrent M1 network) is an $N \times M$ matrix composed of M_E non-negative columns and M_I non-positive columns (thus $M = M_\mathrm{E} + M_\mathrm{I}$). Such a sign-structured decomposition will allow optimal control to be performed through the more realistic feedback architecture shown in Figure 3.14C, with corresponding dynamics of the form:

$$\begin{aligned} \text{M1} \qquad & \tau \frac{d\boldsymbol{x}}{dt} = -\boldsymbol{x}(t) + \boldsymbol{W}\phi\left[\boldsymbol{x}(t)\right] + \overline{\boldsymbol{h}} + \boldsymbol{h}(t) + \tilde{\boldsymbol{u}}^\star + \boldsymbol{K}_{xz}\phi[\boldsymbol{z}(t)] \\ \text{M1-layer 4} \qquad & \boldsymbol{z}(t) = \boldsymbol{K}_{zy}\phi[\boldsymbol{y}(t)] \\ \text{Thal.} \qquad & \boldsymbol{y}(t) = \boldsymbol{K}_{yx}\phi[\boldsymbol{x}(t)] \end{aligned} \tag{3.46}$$

where

$$\tilde{\boldsymbol{u}}^\star = \boldsymbol{x}^\star - (\boldsymbol{W} + \boldsymbol{K})\phi(\boldsymbol{x}^\star) - \overline{\boldsymbol{h}} \tag{3.47}$$

is a condition-dependent steady input given to the network during movement preparation so as to achieve the desired fixed point $\boldsymbol{x}^\star$.

To achieve this decomposition, we note that without loss of generality we can choose $\boldsymbol{K}_{yx} = [\boldsymbol{R}\ \boldsymbol{0}_{N_\mathrm{E} \times N_\mathrm{I}}]$ – where $\boldsymbol{R}$ is a random, element-wise positive $N_\mathrm{E} \times N_\mathrm{E}$ matrix – and apply the algorithm developed in Section 3.4.6 now with $\boldsymbol{\Gamma} = \boldsymbol{K}_{yx}$. This will return an optimal $N \times N_\mathrm{E}$ matrix $\boldsymbol{Z}$ describing feedback from thalamus back to M1, which – as long as $\boldsymbol{R}$ is invertible – will achieve the same minimum cost as if $\boldsymbol{R}$ had been set to $\boldsymbol{I}_{N_\mathrm{E}}$ (as in Section 3.4.6). Here, we simply draw each element of $\boldsymbol{R}$ from Bernoulli(p), i.e. random sparse projections (the magnitude of $\boldsymbol{R}$ does not matter at this stage, as only the product $\boldsymbol{Z}\boldsymbol{\Gamma}$ does; $\boldsymbol{R}$ will be renormalized later below). We now need to decompose this optimal feedback matrix as $\boldsymbol{Z} = \boldsymbol{K}_{xz}\boldsymbol{K}_{zy}$, with the same sign constraints as in Equation (3.45). We approach this via optimization, by minimizing the squared error implied by the decomposition, plus an 2-norm regularizer:

$$\frac{\|\boldsymbol{Z} - \boldsymbol{K}_{xz}\boldsymbol{K}_{zy}\|_\mathrm{F}^2}{\|\boldsymbol{Z}\|_\mathrm{F}^2} + \gamma\left(\|\boldsymbol{K}_{xz}\|_\mathrm{F}^2 + \|\boldsymbol{K}_{zy}\|_\mathrm{F}^2\right) \tag{3.48}$$

We parameterize each element of $\boldsymbol{K}_{xz}$ and $\boldsymbol{K}_{zy}$ as $\pm z^2$, where z is a free parameter to be optimized, and the $\pm$ sign enforces the sign structure written in Equation (3.45). Minimization is achieved using BFGS and typically converges in a few tens of iterations. We note that the product $\boldsymbol{K}_{xz}\boldsymbol{K}_{zy}\boldsymbol{K}_{yx}$ is invariant to any set of rescalings of the individual matrices as long as they cancel out to 1. Thus, after optimization, we re-balance the three matrices such that they have identical Frobenius norms. This is mathematically optional, but ensures that firing rates in M1, thalamus and M1-L4 have approximately the same dynamic range.

Importantly, we find that as long as the number of M1-L4 neurons (M) is chosen sufficiently large, the decomposition of $\boldsymbol{Z}$ that we obtain is almost exact, which implies that the dynamics of Equation (3.46) still achieves optimal anticipatory control of movement under the architectural constraint of Equation (3.38).

Taking into account integration dynamics in thalamus and M1-layer 4

The optimal control solution that we arrived at in Equation (3.46) still relies on instantaneous feedback from cortex back onto itself. However, neurons in the thalamus and in M1's input layer have their own integration dynamics – this will introduce lag around the loop, which must be taken into account when designing the optimal feedback. We therefore include these dynamics:

$$\begin{aligned} \text{M1} \qquad & \tau \frac{d\boldsymbol{x}}{dt} = -\boldsymbol{x}(t) + \boldsymbol{W}\phi\left[\boldsymbol{x}(t)\right] + \overline{\boldsymbol{h}} + h(t) + \tilde{\boldsymbol{u}}^\star + \boldsymbol{K}_{xz}\phi[\boldsymbol{z}(t)] \\ \text{M1-layer 4} \qquad & \tau_z \frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + \boldsymbol{K}_{zy}\phi[\boldsymbol{y}(t)] \\ \text{Thal.} \qquad & \tau_y \frac{d\boldsymbol{y}}{dt} = -\boldsymbol{y} + \boldsymbol{K}_{yx}\phi[\boldsymbol{x}(t)] \end{aligned} \tag{3.49}$$

where the steady input $\tilde{\boldsymbol{u}}^\star$ is again given by Equation (3.47), and $\{\tau_y, \tau_z\}$ are the single-neuron time constants in the thalamus and the cortical input layer. We then seek the optimal connectivity matrices $\{\boldsymbol{K}_{xz}, \boldsymbol{K}_{zy}, \boldsymbol{K}_{yx}\}$ to fulfill the same optimal-control principles as before, namely the minimization of the cost functional in Equation (3.28). In order to do that, we note that the dynamics of $\boldsymbol{x}$ (M1 activity) in the linear regime do not change if the system of differential equations in Equation (3.49) is simplified as

$$\begin{aligned} \text{M1} \qquad & \tau \frac{d\boldsymbol{x}}{dt} = (\boldsymbol{W} - \boldsymbol{I})\boldsymbol{x}(t) + \overline{\boldsymbol{h}} + h(t) + \tilde{\boldsymbol{u}}^\star + \boldsymbol{K}\boldsymbol{z}(t) \\ \text{M1-layer 4} \qquad & \tau_z \frac{d\boldsymbol{z}}{dt} = -\boldsymbol{z} + \boldsymbol{y}(t) \\ \text{Thal.} \qquad & \tau_y \frac{d\boldsymbol{y}}{dt} = -\boldsymbol{y} + \boldsymbol{x}(t) \end{aligned} \tag{3.50}$$

where $\boldsymbol{K} = \boldsymbol{K}_{xz}\boldsymbol{K}_{zy}\boldsymbol{K}_{yx}$ summarizes the three connectivity matrices around the loop into one effective feedback gain matrix. This formulation allows us to combine the steps developed in Section 3.4.6 to find the optimal connectivity matrices.

Specifically, we apply the algorithm of Section 3.4.6 to an augmented system with state matrix

$$\boldsymbol{A}' \triangleq \begin{bmatrix} \boldsymbol{A} & \boldsymbol{0}_{N\times N_\mathrm{E}} & \boldsymbol{0}_{N\times N_\mathrm{E}} \\ (\tau/\tau_y)[\boldsymbol{R}\ \boldsymbol{0}] & -(\tau/\tau_y)\boldsymbol{I}_{N_\mathrm{E}} & \boldsymbol{0}_{N_\mathrm{E}} \\ \boldsymbol{0} & (\tau/\tau_z)\boldsymbol{I}_{N_\mathrm{E}} & -(\tau/\tau_z)\boldsymbol{I}_{N_\mathrm{E}} \end{bmatrix}, \tag{3.51}$$

input matrix

$$\boldsymbol{B}' \triangleq \begin{bmatrix} \boldsymbol{I}_N \\ \boldsymbol{0}_{N_\mathrm{E}\times N} \\ \boldsymbol{0}_{N_\mathrm{E}\times N} \end{bmatrix}, \tag{3.52}$$

quadratic cost weighting matrix

$$\boldsymbol{Q}' \triangleq \begin{bmatrix} \boldsymbol{Q} & \boldsymbol{0}_{N\times N_\mathrm{E}} & \boldsymbol{0}_{N\times N_\mathrm{E}} \\ \boldsymbol{0}_{N_\mathrm{E}\times N} & \boldsymbol{0}_{N_\mathrm{E}\times N_\mathrm{E}} & \boldsymbol{0}_{N_\mathrm{E}\times N_\mathrm{E}} \\ \boldsymbol{0}_{N_\mathrm{E}\times N} & \boldsymbol{0}_{N_\mathrm{E}\times N_\mathrm{E}} & \boldsymbol{0}_{N_\mathrm{E}\times N_\mathrm{E}} \end{bmatrix} \tag{3.53}$$

and feedback input parameterized as

$$\boldsymbol{u} = \boldsymbol{K}\boldsymbol{x} \quad \text{with} \quad \boldsymbol{K} = \boldsymbol{Z}\boldsymbol{\Gamma} = \boldsymbol{Z}\,[\boldsymbol{0}_{N_\mathrm{E}\times N} \quad \boldsymbol{0}_{N_\mathrm{E}\times N_\mathrm{E}} \quad \boldsymbol{I}_{N_\mathrm{E}}] \tag{3.54}$$

In Equation (3.51), the matrix $\boldsymbol{R}$ is again a random matrix of sparse positive connections from M1 to thalamus (cf. Section 3.4.6 above). The optimal $\boldsymbol{Z}$ (Section 3.4.6) corresponds to the product $\boldsymbol{K}_{xz}\boldsymbol{K}_{zy}$, which we can further decompose under sign constraints to recover the individual connectivity matrices $\boldsymbol{K}_{xz}$ and $\boldsymbol{K}_{zy}$.

Disinhibitory action of the basal ganglia

We model the disinhibitory action of the basal ganglia (BG) on thalamic neurons as an ON-OFF switch: to trigger movement, BG become active (BG neurons not explicitly modelled here) and the thalamic neurons are silenced instantly (i.e. $\boldsymbol{y}$ is set to $\boldsymbol{0}$). When this happens, thalamic inputs to M1-L4 vanish and M1-L4 neural activity decays to zero on a time-scale τ_z (see Equation (3.51)). As the activity of L4 neurons decays, these neurons continue to exert an influence on M1 activity through the connectivity matrix $\boldsymbol{K}_{zx}$. This lead to changes in movement-related M1 dynamics, resulting in small movement errors, which we correct post-hoc by re-optimizing the desired initial state $\boldsymbol{x}^\star$ for each movement. From these new desired states, network dynamics evolves—with the additional inputs from M1-L4 neurons after movement onset—to produce accurate hand trajectories. Crucially, unlike what we described in Section 3.4.1, we do not re-optimize the readout matrix $\boldsymbol{C}$ here. This is because the observability Gramian $\boldsymbol{Q}$ and thus the closed-loop controller $\boldsymbol{K}$ depend on $\boldsymbol{C}$: changing the readout matrix $\boldsymbol{C}$ at this stage would cause the $\boldsymbol{K}$ we found to no longer be optimal with respect to $\mathcal{J}$. However, because the closed-loop solution does not depend on the desired fixed points $\boldsymbol{x}^\star$, we can re-optimize $\boldsymbol{x}^\star$ and still be guaranteed that the $\boldsymbol{K}$ that we found remains optimal.

Modelling the effect of photoinhibition

To model photoinhibition in our full circuit (whose dynamics are described by Equation (3.46)), we simply add a constant positive input h_{ph} to a subset of cortical inhibitory neurons chosen randomly, for a duration $T_{\text{ph}} = 400$ ms (see parameters in Table 3.1). This results in an overall decrease in population activity across both excitatory and inhibitory neurons, consistent with the well-known paradoxical effects of adding positive inputs to I cells in inhibition-stabilized networks [128, 156, 190].

In Figure 3.9D, we consider how, after photoinhibition, the activity of the full circuit recovers in three subspaces: the coding subspace (CS), the persistent subspace (PS), and the remaining subspace (RS). To define the CS, we focus on *unperturbed* neural activity in the last 400 ms of movement preparation. For each movement condition, we compute the time-averaged population activity vector in this time window and combine them into a data matrix $\boldsymbol{X}_{\text{unpert}} \in \mathbb{R}^{N\times 8}$. We perform principal components analysis (PCA) on $\boldsymbol{X}_{\text{unpert}}$ and extract an orthonormal basis $\boldsymbol{U}_{\text{CS}}$, which captures 90% of the variance in $\boldsymbol{X}_{\text{unpert}}$. The CS is defined as the subspace spanned by the columns of $\boldsymbol{U}_{\text{CS}}$.

The PS and the RS are defined specifically for each perturbation experiment. In each experiment, we simulate the perturbed dynamics of the network for each movement condition, each with a different random subset of inhibitory neurons being 'photostimulated'. To compute the PS, we consider the same time window that we do for the CS and calculate the *perturbed*, time-averaged population activity vector for each movement. We again collate them into a matrix $\boldsymbol{X}_{\text{pert}} \in \mathbb{R}^{N\times 8}$ and perform PCA on the deviation induced by the perturbations in the given experiment, $\boldsymbol{\Delta} = \boldsymbol{X}_{\text{pert}} - \boldsymbol{X}_{\text{unpert}}$. We extract an orthonormal basis $\boldsymbol{U}_{\text{PS}}$, which captures 90% of the variance in $\boldsymbol{\Delta}$. The PS is defined as the subspace spanned by the columns of $\boldsymbol{U}_{\text{PS}}$. Similarly, to compute the RS, we construct the data matrix $\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{X}_{\text{unpert}} & \boldsymbol{X}_{\text{pert}} \end{bmatrix}$ in each perturbation experiment and orthogonalize it with respect to both $\boldsymbol{U}_{\text{CS}}$ and $\boldsymbol{U}_{\text{PS}}$. We perform PCA on the resulting data matrix and again extract an orthonormal basis $\boldsymbol{U}_{\text{RS}}$ that captures 90% of the variance, which defines the RS. We find that the three subspaces combine to capture 98% of the total variance in $\boldsymbol{Y}$, averaged over 300 independent perturbation experiments.

To examine how activity trajectories recover in the three subspaces, we project the perturbed and unperturbed activity onto $\boldsymbol{U}_{\text{CS}}$, $\boldsymbol{U}_{\text{PS}}$, and $\boldsymbol{U}_{\text{RS}}$ and calculate the magnitude of the deviation in each subspace. This is averaged over all movements and 300 independent perturbation experiments, which is what is shown in Figure 3.9D.

3.4.7 Universal performance of naive feedforward control

In this section, we show that for a linear system solving a motor task, the anticipatory control cost $\mathcal{J}$ under the naive feedforward strategy only depends on the movements that are generated as part of the task.

Consider a network characterized by a state matrix $\boldsymbol{A}$ and generating a given movement using readout weights $\boldsymbol{C}$ and an initial condition $\boldsymbol{x}^\star$ using the following movement-epoch dynamics:

$$\tau \frac{d\boldsymbol{x}}{dt} = \boldsymbol{A}\boldsymbol{x}(t) + \overline{\boldsymbol{h}} \tag{3.55}$$

where $\overline{\boldsymbol{h}} = -\boldsymbol{A}\boldsymbol{x}_{\text{sp}}$ sets the spontaneous fixed point at $\boldsymbol{x}_{\text{sp}}$. These are the same dynamics as Equation (3.7) but excluding $\boldsymbol{h}(t)$ as explained in Section 3.4.1. When the feedforward strategy is used for preparatory control, a constant preparatory input $\boldsymbol{u}^\star$ (Equation (3.26)) is added to the r.h.s. of Equation (3.55) that instantaneously switches the network's fixed point from $\boldsymbol{x}_{\text{sp}}$ to $\boldsymbol{x}^\star$. Thus,

$$\boldsymbol{x}(t) = \boldsymbol{x}^\star + e^{\frac{t}{\tau}\boldsymbol{A}}(\boldsymbol{x}_{\text{sp}} - \boldsymbol{x}^\star). \tag{3.56}$$

Now, let $\boldsymbol{x}(t, t')$ be the activity of the network at time $t + t'$, where t marks the end of preparation and the beginning of the movement. At time t, the constant preparatory input $\boldsymbol{u}^\star$ is withdrawn, causing the fixed point of the dynamics to switch back to $\boldsymbol{x}_{\text{sp}}$. Therefore,

$$\boldsymbol{x}(t, t') = \boldsymbol{x}_{\text{sp}} + e^{\frac{t'}{\tau}\boldsymbol{A}}(e^{\frac{t}{\tau}\boldsymbol{A}} - I)(\boldsymbol{x}_{\text{sp}} - \boldsymbol{x}^\star) \tag{3.57}$$

and the corresponding output torques as $\boldsymbol{C}\boldsymbol{x}(t, t')$. To compute the corresponding *error* in torques, $\boldsymbol{m}(t, t') - \boldsymbol{m}^\star(t')$, we note that the system produces the target movements with no error in the limit of infinite preparation time ($t \to \infty$). Thus, the momentary error at time $t + t'$ is given by

$$\begin{aligned} \boldsymbol{m}(t, t') - \boldsymbol{m}^\star(t') &= \boldsymbol{C}\left[\boldsymbol{x}(t, t') - \boldsymbol{x}(\infty, t')\right] && (3.58) \\ &= \boldsymbol{C}e^{\frac{t+t'}{\tau}\boldsymbol{A}}(\boldsymbol{x}_{\text{sp}} - \boldsymbol{x}^\star) && (3.59) \end{aligned}$$

where we have also used the fact that $\boldsymbol{C}\boldsymbol{x}_{\text{sp}} = 0$. Importantly, it is easy to show that Equation (3.59) is in fact equal to the target output torque $t + t'$ seconds after movement onset, i.e. $\boldsymbol{m}^\star(t + t') = \boldsymbol{C}\boldsymbol{x}(\infty, t + t')$. In summary, under the naive feedforward control strategy, motor errors following insufficiently long preparation only depend on the target movements, but not on the actual (linear) system meant to achieve them. *A fortiori*, since the control cost $\mathcal{J}$ (Equation (3.28)) is defined based on output errors in torques, it must also be the same for any network that has been successfully trained to produce the target torques (in the limit of long

preparation). We calculated this cost to be

$$\mathcal{J}_{\text{naive}} = \frac{1}{\tau^2} \iint_0^\infty dt\, dt' \, \|\boldsymbol{m}^\star(t+t')\|^2. \tag{3.60}$$

3.4.8 Alignment index under naive feedforward control

Early preparatory activity under the naive feedforward strategy can be shown to be the negative image of movement-epoch activity, up to a constant offset. Consider a network solution to the reaching task characterized by linear dynamics with a state matrix $\boldsymbol{A}$ and a set of initial conditions $\{\boldsymbol{x}_k^\star\}$. During the preparatory epoch for movement k, the network activity evolves according to

$$\tau \frac{d\boldsymbol{x}}{dt} = \boldsymbol{A}\boldsymbol{x}(t) + \bar{\boldsymbol{h}} + \boldsymbol{u}_k^\star \tag{3.61}$$

where $\boldsymbol{u}_k^\star$ sets the fixed point of the preparatory dynamics at $\boldsymbol{x}_k^\star$. Thus, preparatory activity $\boldsymbol{x}_k^{\text{p}}(t)$ for a movement k is given by:

$$\boldsymbol{x}_k^{\text{p}}(t) = \boldsymbol{x}_k^\star + e^{\frac{t}{\tau}\boldsymbol{A}}(\boldsymbol{x}_{\text{sp}} - \boldsymbol{x}_k^\star). \tag{3.62}$$

Movement-epoch dynamics, on the other hand, obey:

$$\tau \frac{d\boldsymbol{x}}{dt} = \boldsymbol{A}\boldsymbol{x}(t) + \bar{\boldsymbol{h}} + \boldsymbol{h}(t) \tag{3.63}$$

where $\bar{\boldsymbol{h}} = -\boldsymbol{A}\boldsymbol{x}_{\text{sp}}$ sets the fixed point at $\boldsymbol{x}_{\text{sp}}$ and $\boldsymbol{h}(t)$ is the condition-independent input ($\boldsymbol{h}(t) = \boldsymbol{0}$ for the 'full' and 'low-rank' networks). Assuming the network activity successfully reaches $\boldsymbol{x}_k^\star$ during the preparation epoch, movement activity $\boldsymbol{x}_k^{\text{m}}(t)$ is given by

$$\boldsymbol{x}_k^{\text{m}}(t) = \boldsymbol{x}_{\text{sp}} + e^{\frac{t}{\tau}\boldsymbol{A}}(\boldsymbol{x}_k^\star - \boldsymbol{x}_{\text{sp}}) + \boldsymbol{q}(t), \tag{3.64}$$

where

$$\boldsymbol{q}(t) = \int_0^t e^{\frac{t-t'}{\tau}\boldsymbol{A}} \boldsymbol{h}(t') dt' \tag{3.65}$$

is the contribution of the condition-independent external drive $\boldsymbol{h}(t)$. Comparing Equation (3.63) and Equation (3.64), we find that $\boldsymbol{x}_k^{\text{p}}$ is the negative image of $\boldsymbol{x}_k^{\text{m}}$, up to a constant offset, and condition-independent temporal variations:

$$\boldsymbol{x}_k^{\text{p}}(t) = -\boldsymbol{x}_k^{\text{m}}(t) + \boldsymbol{x}_k^\star + \boldsymbol{x}_{\text{sp}} + \boldsymbol{q}(t). \tag{3.66}$$

In computing the alignment index, Elsayed et al. [37] first removed the mean across condition at every time point, and we do the same here. Therefore,

$$\hat{\boldsymbol{x}}_k^{\text{p}}(t) = -\hat{\boldsymbol{x}}_k^{\text{m}}(t) + \hat{\boldsymbol{x}}_k^\star \tag{3.67}$$

where $\hat{\cdot}$ denotes deviation from the condition mean. This explains the high alignment index of all network classes under the naive feedforward strategy (Figure 3.7C).

3.5 Quantification and statistical analysis

3.5.1 Network measures

Participation ratio To estimate the dimensionality of subspaces, we use the "participation ratio" [51], calculated based on the eigenvalue spectrum $\sigma_1, \sigma_2, \cdots, \sigma_N$ of the relevant symmetric, positive-definite matrix as

$$\frac{(\sum_i \sigma_i)^2}{\sum_i \sigma_i^2}. \tag{3.68}$$

Nonnormality In Figure 3.5E, we define the nonnormality of a network with synaptic connectivity matrix $\boldsymbol{W}$ as

$$\frac{\|\boldsymbol{W}\|_{\mathrm{F}}^2 - \|\boldsymbol{\Lambda}\|_{\mathrm{F}}^2}{\|\boldsymbol{W}\|_{\mathrm{F}}^2} \tag{3.69}$$

as proposed by Murphy and Miller [121], where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues of $\boldsymbol{W}$.

$\mathcal{H}_2$ norm In Figure 3.5F, we define the $\mathcal{H}_2$ norm of a network as $\sqrt{\mathrm{Tr}\,(\boldsymbol{\mathcal{W}_o})}$, where the full-state observability Gramian $\boldsymbol{\mathcal{W}}_o$ satisfies the Lyapunov equation $\boldsymbol{A}^T\boldsymbol{\mathcal{W}}_o + \boldsymbol{\mathcal{W}}_o\boldsymbol{A} + \boldsymbol{I} = \boldsymbol{0}$ and $\boldsymbol{A} = -\boldsymbol{I} + \boldsymbol{W}$.

Prospective motor potency The prospective motor potency of a state space direction $\boldsymbol{d}$ (with $\|\boldsymbol{d}\| = 1$) is $\boldsymbol{d}^T\boldsymbol{Q}\boldsymbol{d}$, where $\boldsymbol{Q}$ satisfies Equation (3.25). This quantifies the prospective motor error induced by a deviation $\boldsymbol{x}^\star + \boldsymbol{d}$ of the final preparatory state away from $\boldsymbol{x}^\star$. In Figure 3.3D, we plot the top 20 eigenvalues of $\boldsymbol{Q}$, i.e. the motor potency of the 20 most potent directions (the corresponding eigenvectors of $\boldsymbol{Q}$). The prospective motor potency of a K-dimensional subspace $\mathcal{S}$, spanned by the unit vectors $\boldsymbol{d}_1, \cdots, \boldsymbol{d}_K$, is given by

$$\frac{1}{K}\sum_{i=1}^{K} \boldsymbol{d}_i^T\boldsymbol{Q}\boldsymbol{d}_i. \tag{3.70}$$

Amplification factor In Figure 3.7D, for each movement k we define the amplification factor in the model as $\|(\boldsymbol{x}(t) - \boldsymbol{x}_{\mathrm{sp}})\|/\|\boldsymbol{x}_k^\star - \boldsymbol{x}_{\mathrm{sp}}\|$, where $\boldsymbol{x}(t)$ is the movement-epoch activity evolving according to Equation (3.7) with $\boldsymbol{h}(t) = 0$, $\boldsymbol{u}(t) = 0$, and $\boldsymbol{x}(0) = \boldsymbol{x}_k^\star$. Description of how we define the amplification factor for the monkeys can be found below.

3.5.2 Neural data analysis

We analyzed neural recordings of two monkeys (J and N) performing a delayed reaching task (data courtesy of Mark Churchland, Matt Kaufman and Krishna Shenoy). Both the task and dataset have been described in detail previously [22]. Briefly, the two monkeys performed center-out reaches on a fronto-parallel screen. At the beginning of each trial, they fixated on the centre of the screen for some time, after which a target appeared on the screen. A variable delay period (0–1000 ms) ensued, followed by a go cue instructing the monkeys to reach towards the target. In this paper, we analyzed nine movement conditions, corresponding to the straight reaches that were most similar to the ones we modelled (Figure 3.2B). Moreover, we restricted our analysis to the trials with delay periods longer than 400 ms.

Recordings were made in the dorsal premotor and primary motor areas. We preprocessed the spike trains of 123 neurons for monkey J and 221 neurons for monkey N, following the same procedure outlined in Churchland et al. [21]. Briefly, we computed the average firing rates for each movement condition, further smoothed using a 20 ms Gaussian filter. Firing rates were computed separately for the delay and movement periods, time-locked to target and movement onset respectively; this is necessary because of variable delay periods and reaction times.

There is some subtletly in our definition of the time of movement onset in the model, when we compare its activity to monkey data. In the model, movement begins at the same time neural activity begins to undergo rapid changes, i.e. as soon as control inputs are removed. However, in the two monkeys, such rapid changes in neural activity occur roughly 100 ms before movement begins. We attribute such delays in movement to delays in downstream motor processes not considered in our model. Therefore, to align the temporal profile of neural activity in the model and the data, we define the time of "movement onset" in the model to be 100 ms after the control inputs are removed. We perform such temporal alignment in all the data and model comparisons shown in Figure 3.2, Figure 3.4, and Figure 3.7.

Overlap between preparatory end-states

We calculated the Pearson correlation across neurons between the preparatory end-states in both model and monkey data for all reaches. Preparatory end-states are defined as the activity states reached at the end of movement preparation (monkey activity aligned to the go cue). In both model and monkey data, preparatory end-states are similar (Figure 3.2B, bottom) for reaches with similar hand trajectories (hand trajectories Figure 3.2B, top), but negatively correlated for more distant movements.

Amplification factor

To compute the amplification factor for the two monkeys in Figure 3.7D, we consider firing rates in a 400 ms window starting 250 ms prior to movement onset. We remove the mean across conditions and compute for each reach condition, the manitude of the population activity vector, normalized by its magnitude at the start of this epoch. We then average this across all reach conditions. The amplification factor quantifies the expansion of the population activity across neurons during the movement epoch.

jPCA

We used the method described in Churchland et al. [21] to identify state-space directions in which activity trajectories rotate most strongly. Briefly, we used numerical optimization to fit a skew-symmetric linear dynamical system of the form $\dot{\boldsymbol{x}} = \boldsymbol{S}\boldsymbol{x}$ that best captures the population activity in a 400 ms window starting 220 ms before movement onset. We projected population activity in this window onto a plane spanned by the top two eigenvectors of $\boldsymbol{S}$ (Figure 3.2B).

Alignment index

To calculate the alignment index, we closely followed the methods described in Elsayed et al. [37]. The alignment index is defined as the (normalized) percentage of across-condition variance during movement captured by the top K principal components (PCs) of the preparatory activity:

$$\mathrm{Tr}\left(\frac{\boldsymbol{D}_{\mathrm{move}}^{T}\boldsymbol{C}_{\mathrm{prep}}\boldsymbol{D}_{\mathrm{move}}}{\sum_{i=1}^{K}\sigma_{i,\mathrm{prep}}^{2}}\right) \tag{3.71}$$

where the K columns of $\boldsymbol{D}_{\mathrm{move}}$ are the top K principal components of move. activity ("move-PCs"), $\boldsymbol{C}_{\mathrm{prep}}$ is the covariance matrix of prep. activity, and $\sigma_{i,\mathrm{prep}}^{2}$ is the prep. activity variance captured by the i^{th} prep-PC. We choose K such that K prep-PCs captures 80% of the variance in prep. activity. Here, we define prep. activity as the delay-period activity during a 300 ms window starting 150 ms after target onset; the activity is calculated time-locked to target onset. Similarly, move. activity is defined as activity during a 300 ms window starting 50 ms prior to movement onset; the activity is calculated using firing rates time-locked to movement onset.

Methods for calculating the control of the alignment index are described in detail in the Supplementary Material of Elsayed et al. [37] and are not reproduced here. For the model, the alignment index is calculated in the same way as for the neural data.

Canonical-correlation analysis

To compare model and monkey activity, we performed canonical-correlation analysis (CCA) on activity in a time window starting 400 ms before and ending 400 ms after movement onset (see discussion above for nuance in defining the time of movement onset in the model). To avoid overfitting to noise in CCA [142, 178], we first reduced the dimensionality of the two data sets, by projecting activity onto the top 21 (monkey J), 21 (monkey N), and 14 (model) principal components; the number of principal components are chosen to capture 95% of the across-condition activity variance in the two datasets. We then calculated the canonical correlations between the two reduced data sets, using the numerically stable algorithm described in Press [141]. We found that monkey and model activity are similar across time and reaches, with a high average canonical correlation (Figure 3.2C-E). We obtained similar results when we varied the number of principal components kept in the two data sets, which in turn varied the number of canonical variables.

Chapter 4

Automatic differentiation of Sylvester, Lyapunov, and algebraic Riccati equations

Summary

Sylvester, Lyapunov, and algebraic Riccati equations are the bread and butter of control theorists. They are used to compute infinite-horizon Gramians, solve optimal control problems in continuous or discrete time, and design observers. While popular numerical computing frameworks (e.g., `scipy`) provide efficient solvers for these equations, these solvers are still largely missing from most automatic differentiation libraries. Here, we derive the forward and reverse-mode derivatives of the solutions to all three types of equations, and showcase their application on an inverse control problem.

4.1 Introduction

In recent years, automatic differentiation (AD) has become the backbone of most modern machine learning applications and an important tool for scientific enquiry [10]. The success of AD owes in part to the myriad of forward and reverse-mode derivatives that have been derived for matrix/tensor operations including most known linear operators and factorizations [56]. These are commonly available in popular automatic differentiation libraries such as PyTorch [136], Tensorflow [136], Jax [11], Diffsharp [10], and Zygote [77].

In this technical note, we derive the forward and reverse-mode gradients for solutions to three important types of matrix equations used extensively in control theory: Sylvester, Lyapunov, and algebraic Riccati equations. Although efficient solvers are readily available in popular scientific programming libraries, to the best of our knowledge they remain missing from most automatic differentiation packages (though CasADi seems to provide Lyapunov solvers; 5, 57). We have added these solvers to Owl [198], a numerical library written in OCaml with a full-featured AD module, and we hope that this technical note will enable rapid integration in other popular AD packages.

4.2 Preliminaries

We use bold upper-case letters to denote matrices and bold lower-case letters to denote vectors. We use $\boldsymbol{X}^T$, $\boldsymbol{X}^{-1}$ and $\mathrm{tr}(\boldsymbol{X})$ to denote the transpose, inverse, and trace of a matrix $\boldsymbol{X} \in \mathbb{R}^{n \times n}$. Following Giles [56], we use $\overline{\boldsymbol{X}}$ to denote the adjoint of matrix $\boldsymbol{X}$ w.r.t. a scalar output ℓ, i.e. $\partial \ell / \partial \boldsymbol{X}$. We also use $\dot{\boldsymbol{X}}$ to denote the tangent of a matrix $\boldsymbol{X}$ w.r.t. some scalar input s, i.e. $\partial \boldsymbol{X} / \partial s$.

Consider a differentiable solver $\boldsymbol{P} = f(\boldsymbol{A}, \boldsymbol{B}, \ldots)$. In forward mode, the input tangents $(\dot{\boldsymbol{A}}, \dot{\boldsymbol{B}}, \ldots)$ are known and used to calculate $\dot{\boldsymbol{P}}$. In reverse mode, the solution's adjoint $\overline{\boldsymbol{P}}$ is known and used to update the input's adjoints $(\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \ldots)$. This note consists mainly in deriving the rules by which these tangents and adjoints should be computed for various types of matrix equation solvers. All our results are summarized in Tables 4.1 and 4.2.

4.3 Sylvester equations

4.3.1 Continuous time Sylvester equation

The continuous time Sylvester equation takes the form

$$\boldsymbol{A}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{B} + \boldsymbol{C} = \boldsymbol{0}, \tag{4.1}$$

where $\boldsymbol{A}$, $\boldsymbol{P}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ are all square matrices that have the same dimensions. There is a unique solution $\boldsymbol{P}$ that satisfies Equation (4.1) for a given $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ if and only if $\boldsymbol{A}$ and $\boldsymbol{B}$ have no common eigenvalue. As we shall see in the following, the matrix derivatives for the solution to the Sylvester equation is also unique under the same condition on the spectrum of $\boldsymbol{A}$ and $\boldsymbol{B}$.

Forward mode Taking the differential on both sides of Equation (4.1), we have

$$\boldsymbol{A}\,d\boldsymbol{P} + d\boldsymbol{P}\,\boldsymbol{B} + (d\boldsymbol{A}\,\boldsymbol{P} + \boldsymbol{P}\,d\boldsymbol{B} + d\boldsymbol{C}) = \boldsymbol{0}. \tag{4.2}$$

The forward mode derivative $\dot{\boldsymbol{P}}$ can thus be computed by solving another continuous time Sylvester equation involving the tangents $\dot{\boldsymbol{A}}$, $\dot{\boldsymbol{B}}$, and $\dot{\boldsymbol{C}}$:

$$\boldsymbol{A}\,\dot{\boldsymbol{P}} + \dot{\boldsymbol{P}}\,\boldsymbol{B} + (\dot{\boldsymbol{A}}\,\boldsymbol{P} + \boldsymbol{P}\,\dot{\boldsymbol{B}} + \dot{\boldsymbol{C}}) = \boldsymbol{0}. \tag{4.3}$$

Reverse mode To derive the adjoints $\overline{\boldsymbol{A}}, \overline{\boldsymbol{B}}, \overline{\boldsymbol{C}}$, we consider the Lagrangian

$$\mathcal{L}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{P}) = \ell(\boldsymbol{P}) + \operatorname{tr}(\boldsymbol{S}^T(\boldsymbol{A}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{B} + \boldsymbol{C})), \tag{4.4}$$

where $\ell(\boldsymbol{P})$ is the scalar output w.r.t. which we wish to compute adjoints, and $\boldsymbol{S}$ is a matrix of Lagrange multipliers used to enforce Equation (4.1). Taking partial derivatives on both sides of the Lagrangian and setting them to zero, we obtain:

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{P}} = \frac{\partial \ell(\boldsymbol{P})}{\partial \boldsymbol{P}} + \boldsymbol{A}^T\boldsymbol{S} + \boldsymbol{S}\boldsymbol{B}^T, \tag{4.5}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}} = \boldsymbol{A}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{B} + \boldsymbol{C}, \tag{4.6}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{A}} = \boldsymbol{S}\boldsymbol{P}^T, \tag{4.7}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{B}} = \boldsymbol{P}^T\boldsymbol{S}, \tag{4.8}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{C}} = \boldsymbol{S}. \tag{4.9}$$

Note that Equation (4.6) does indeed enforce that $\boldsymbol{P}$ be the solution of Equation (4.1). Identifying $\overline{\boldsymbol{P}} = \partial \ell / \partial \boldsymbol{P}$, we find that the Lagrange multiplier $\boldsymbol{S}$ satisfies another Sylvester equation

$$\boldsymbol{A}^T\boldsymbol{S} + \boldsymbol{S}\boldsymbol{B}^T + \overline{\boldsymbol{P}} = \boldsymbol{0}. \tag{4.10}$$

After computing $\boldsymbol{S}$, and noting that when $\boldsymbol{P}$ solves Equation (4.6) then $\mathcal{L} = \ell$, it is straightforward to compute the rest of the adjoints[1] :

$$\overline{\boldsymbol{A}} = \boldsymbol{S}\boldsymbol{P}^T, \qquad \overline{\boldsymbol{B}} = \boldsymbol{P}^T\boldsymbol{S}, \qquad \overline{\boldsymbol{C}} = \boldsymbol{S}. \tag{4.11}$$

4.3.2 Discrete time Sylvester equation

We follow a similar approach as above to differentiate through the discrete time Sylvester equation, which takes the form

$$\boldsymbol{A}\boldsymbol{P}\boldsymbol{B} - \boldsymbol{P} + \boldsymbol{C} = \boldsymbol{0}, \tag{4.12}$$

[1] One can formally show that $\overline{\boldsymbol{X}} = \partial \mathcal{L} / \partial \boldsymbol{X}$ for $\boldsymbol{X} \in \{\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}\}$ by applying the implicit function theorem (see 144 and references therein).

where $\boldsymbol{A}$, $\boldsymbol{P}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ are all square matrices that have the same dimensions. There is a unique solution to Equation (4.12) if and only if $\sigma_i(\boldsymbol{A})\sigma_j(\boldsymbol{B}) \neq 1$ for all i, j, where $\{\sigma_i(\boldsymbol{X})\}$ denotes the eigenvalue spectrum of a matrix $\boldsymbol{X}$.

Forward mode Taking the differential on both sides of Equation (4.12), we have

$$\boldsymbol{A}\,d\boldsymbol{P}\,\boldsymbol{B} - d\boldsymbol{P} + (d\boldsymbol{A}\,\boldsymbol{P}\boldsymbol{B} + \boldsymbol{A}\boldsymbol{P}\,d\boldsymbol{B} + d\boldsymbol{C}) = \boldsymbol{0}. \tag{4.13}$$

The tangent of $\boldsymbol{P}$ is thus computed by solving another discrete time Sylvester equation:

$$\boldsymbol{A}\,\dot{\boldsymbol{P}}\,\boldsymbol{B} - \dot{\boldsymbol{P}} + (\dot{\boldsymbol{A}}\,\boldsymbol{P}\boldsymbol{B} + \boldsymbol{A}\boldsymbol{P}\,\dot{\boldsymbol{B}} + \dot{\boldsymbol{C}}) = \boldsymbol{0}. \tag{4.14}$$

Reverse mode To derive the adjoints $\overline{\boldsymbol{A}}$, $\overline{\boldsymbol{B}}$ and $\overline{\boldsymbol{C}}$, we consider the Lagrangian

$$\mathcal{L}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{P}, \boldsymbol{S}) = \ell(\boldsymbol{P}) + \mathrm{tr}(\boldsymbol{S}^T(\boldsymbol{A}\boldsymbol{P}\boldsymbol{B} - \boldsymbol{P} + \boldsymbol{C})). \tag{4.15}$$

Taking partial derivatives on both sides of the Lagrangian and setting them to zero, we get:

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{P}} = \frac{\partial f(\boldsymbol{P})}{\partial \boldsymbol{P}} + \boldsymbol{A}^T\boldsymbol{S}\boldsymbol{B}^T - \boldsymbol{S}, \tag{4.16}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}} = \boldsymbol{A}\boldsymbol{P}\boldsymbol{B} - \boldsymbol{P} + \boldsymbol{C}, \tag{4.17}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{A}} = \boldsymbol{S}\boldsymbol{B}^T\boldsymbol{P}^T, \tag{4.18}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{B}} = \boldsymbol{P}^T\boldsymbol{A}^T\boldsymbol{S}, \tag{4.19}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{C}} = \boldsymbol{S}. \tag{4.20}$$

Identifying $\overline{\boldsymbol{P}} = \partial f(\boldsymbol{P})/\partial \boldsymbol{P}$, we find that the Lagrange multiplier $\boldsymbol{S}$ satisfies another discrete time Sylvester equation

$$\boldsymbol{A}^T\boldsymbol{S}\boldsymbol{B}^T - \boldsymbol{S} + \overline{\boldsymbol{P}} = \boldsymbol{0}. \tag{4.21}$$

The adjoints are given by

$$\overline{\boldsymbol{A}} = \boldsymbol{S}\boldsymbol{B}^T\boldsymbol{P}^T, \quad \overline{\boldsymbol{B}} = \boldsymbol{P}^T\boldsymbol{A}^T\boldsymbol{S}, \quad \overline{\boldsymbol{C}} = \boldsymbol{S}. \tag{4.22}$$

4.4 Lyapunov equations

4.4.1 Continuous time Lyapunov equation

The continuous time Lyapunov equation is given by

$$\boldsymbol{A}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}^T + \boldsymbol{Q} = \boldsymbol{0}, \tag{4.23}$$

where $\boldsymbol{A}$, $\boldsymbol{P}$, and $\boldsymbol{Q}$ are square matrices that have the same dimensions. Note that the Lyapunov equation is a special case of the Sylvester equation in Equation (4.1), with $\boldsymbol{B} = \boldsymbol{A}^T$ and $\boldsymbol{C} = \boldsymbol{Q}$. In control theory, we are most commonly concerned with solutions to the Lyapunov equation when $\boldsymbol{Q}$ is a symmetric matrix, which also results in a symmetric $\boldsymbol{P}$. Below, we derive $\overline{\boldsymbol{P}}$ and $\dot{\boldsymbol{P}}$ in the general case, and specialize to the case where $\boldsymbol{Q}$ is symmetric afterwards.

Forward mode Taking the differential on both sides of Equation (4.23), we get

$$\boldsymbol{A}\,\boldsymbol{dP} + \boldsymbol{dP}\,\boldsymbol{A}^T + (\boldsymbol{dA}\,\boldsymbol{P} + \boldsymbol{P}\,\boldsymbol{dA}^T + \boldsymbol{dQ}) = \boldsymbol{0}. \tag{4.24}$$

Thus, $\dot{\boldsymbol{P}}$ satisfies the continuous time Lyapunov equation:

$$\boldsymbol{A}\dot{\boldsymbol{P}} + \dot{\boldsymbol{P}}\boldsymbol{A}^T + (\dot{\boldsymbol{A}}\boldsymbol{P} + \boldsymbol{P}\dot{\boldsymbol{A}}^T + \dot{\boldsymbol{Q}}) = \boldsymbol{0}. \tag{4.25}$$

Reverse mode In reverse-mode, we consider the Lagrangian

$$\mathcal{L}(\boldsymbol{A}, \boldsymbol{Q}, \boldsymbol{P}, \boldsymbol{S}) = \ell(\boldsymbol{P}) + \operatorname{tr}(\boldsymbol{S}^T(\boldsymbol{A}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}^T + \boldsymbol{Q})), \tag{4.26}$$

where $\boldsymbol{S}$ is the Lagrange multiplier and $\ell(\boldsymbol{P})$ is some loss downstream that is a function of $\boldsymbol{P}$. Taking partial derivatives and setting them to zero, we get the following equations

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{P}} = \frac{\partial \ell(\boldsymbol{P})}{\partial \boldsymbol{P}} + \boldsymbol{A}^T\boldsymbol{S} + \boldsymbol{S}\boldsymbol{A} \tag{4.27}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}} = \boldsymbol{A}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A}^T + \boldsymbol{Q} \tag{4.28}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{Q}} = \boldsymbol{S} \tag{4.29}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{A}} = \boldsymbol{S}\boldsymbol{P}^T + \boldsymbol{S}^T\boldsymbol{P}. \tag{4.30}$$

In reverse-mode, $\partial \ell(\boldsymbol{P})/\partial \boldsymbol{P} = \overline{\boldsymbol{P}}$, which is known in the reverse pass. We can thus compute $\boldsymbol{S}$, by solving the Lyapunov equation

$$\overline{\boldsymbol{P}} + \boldsymbol{A}^T\boldsymbol{S} + \boldsymbol{S}\boldsymbol{A} = \boldsymbol{0}. \tag{4.31}$$

This then gives:

$$\overline{\boldsymbol{A}} = \boldsymbol{S}\boldsymbol{P}^T + \boldsymbol{S}^T\boldsymbol{P} \quad \text{and} \quad \overline{\boldsymbol{Q}} = \boldsymbol{S} \tag{4.32}$$

using the equations above. If $\boldsymbol{Q}$ is a symmetric matrix, as is the case in most control applications, then $\boldsymbol{P}$ and $\boldsymbol{S}$ are also symmetric. In addition, if $\boldsymbol{Q}$ is positive semi-definite and all eigenvalues of A have negative real parts, then $\boldsymbol{P}$ and $\boldsymbol{S}$ are positive semi-definite.

4.4.2 Discrete time Lyapunov equation

The discrete time Lyapunov equation is given by

$$\boldsymbol{A}\boldsymbol{P}\boldsymbol{A}^T - \boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0}, \tag{4.33}$$

Forward mode Taking the differential on both sides of Equation (4.33), we get another discrete time Lyapunov equation:

$$\boldsymbol{A}\,d\boldsymbol{P}\,\boldsymbol{A}^T - d\boldsymbol{P} + (d\boldsymbol{A}\,\boldsymbol{P}\boldsymbol{A}^T + \boldsymbol{A}\boldsymbol{P}\,d\boldsymbol{A}^T + d\boldsymbol{Q}) = \boldsymbol{0}. \tag{4.34}$$

Thus, we can compute $\dot{\boldsymbol{P}}$ by solving:

$$\boldsymbol{A}\dot{\boldsymbol{P}}\boldsymbol{A}^T - \dot{\boldsymbol{P}} + (\dot{\boldsymbol{A}}\boldsymbol{P}\boldsymbol{A}^T + \boldsymbol{A}\boldsymbol{P}\dot{\boldsymbol{A}}^T + \dot{\boldsymbol{Q}}) = \boldsymbol{0}. \tag{4.35}$$

Reverse mode In reverse-mode, we consider the following Lagrangian:

$$\mathcal{L}(\boldsymbol{A}, \boldsymbol{Q}, \boldsymbol{P}, \boldsymbol{S}) = \ell(\boldsymbol{P}) + \operatorname{tr}(\boldsymbol{S}^T(\boldsymbol{A}\boldsymbol{P}\boldsymbol{A}^T - \boldsymbol{P} + \boldsymbol{Q})). \tag{4.36}$$

Taking partial derivatives and setting them to zero, we get:

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{P}} = \frac{\partial \ell(\boldsymbol{P})}{\partial \boldsymbol{P}} + \boldsymbol{A}^T\boldsymbol{S}\boldsymbol{A} - \boldsymbol{S} \tag{4.37}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}} = \boldsymbol{A}\boldsymbol{P}\boldsymbol{A}^T - \boldsymbol{P} + \boldsymbol{Q} \tag{4.38}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{Q}} = \boldsymbol{S} \tag{4.39}$$

$$\boldsymbol{0} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{A}} = \boldsymbol{S}\boldsymbol{A}\boldsymbol{P}^T + \boldsymbol{S}^T\boldsymbol{A}\boldsymbol{P}. \tag{4.40}$$

In reverse-mode, $\partial \ell(\boldsymbol{P})/\partial \boldsymbol{P} = \overline{\boldsymbol{P}}$. We can thus compute $\boldsymbol{S}$, by solving the discrete time Lyapunov equation

$$\overline{\boldsymbol{P}} + \boldsymbol{A}^T\boldsymbol{S}\boldsymbol{A} - \boldsymbol{S} = \boldsymbol{0}. \tag{4.41}$$

This then gives:

$$\overline{\boldsymbol{A}} = \boldsymbol{S}\boldsymbol{A}\boldsymbol{P}^T + \boldsymbol{S}^T\boldsymbol{A}\boldsymbol{P} \quad \text{and} \quad \overline{\boldsymbol{Q}} = \boldsymbol{S}. \tag{4.42}$$

If $\boldsymbol{Q}$ is symmetric, then so are $\boldsymbol{P}$ and $\boldsymbol{S}$, leading to $\overline{\boldsymbol{A}} = 2\boldsymbol{S}\boldsymbol{P}\boldsymbol{A}$.

4.5 Algebraic Riccati equations

4.5.1 Continuous time algebraic Riccati equation

The continuous time algebraic Riccati equation (CARE) is given by

$$\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0}, \tag{4.43}$$

where $\boldsymbol{Q}$ and $\boldsymbol{R}$ are symmetric matrices and $\boldsymbol{R}$ is positive-definite. In most applications, the parameters of this equation will satisfy the conditions that guarantee the existence of a unique (symmetric) solution $\boldsymbol{P}$. Here, we derive $\dot{\boldsymbol{P}}$ and $\overline{\boldsymbol{P}}$ assuming that these conditions are met, so that $\boldsymbol{P} = \boldsymbol{P}^T$.

Forward mode Taking the differential on both sides of Equation (4.43), we obtain a Lyapunov equation:

$$\boldsymbol{dP}\,\tilde{\boldsymbol{A}} + \tilde{\boldsymbol{A}}^T\,\boldsymbol{dP} + \left(\boldsymbol{dZ} + \boldsymbol{dZ}^T + \boldsymbol{dQ} + \boldsymbol{K}^T\,\boldsymbol{dR}\,\boldsymbol{K}\right) = \boldsymbol{0}, \tag{4.44}$$

where $\tilde{\boldsymbol{A}} = \boldsymbol{A} - \boldsymbol{B}\boldsymbol{K}$, $\boldsymbol{K} = \boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P}$, and $\boldsymbol{dZ} = \boldsymbol{P}(\boldsymbol{dA} - \boldsymbol{dB}\,\boldsymbol{K})$. We can thus compute $\dot{\boldsymbol{P}}$ by solving the continuous time Lyapunov equation:

$$\dot{\boldsymbol{P}}\tilde{\boldsymbol{A}} + \tilde{\boldsymbol{A}}^T\dot{\boldsymbol{P}} + \left(\dot{\boldsymbol{Z}} + \dot{\boldsymbol{Z}}^T + \dot{\boldsymbol{Q}} + \boldsymbol{K}^T\dot{\boldsymbol{R}}\boldsymbol{K}\right) = \boldsymbol{0} \tag{4.45}$$

where $\dot{\boldsymbol{Z}} = \boldsymbol{P}(\dot{\boldsymbol{A}} - \dot{\boldsymbol{B}}\,\boldsymbol{K})$.

Reverse mode We consider the Lagrangian

$$\mathcal{L}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{P}, \boldsymbol{S}) = \ell(\boldsymbol{P}) + \operatorname{tr}(\boldsymbol{S}^T(\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P} + \boldsymbol{Q})), \tag{4.46}$$

where $\ell(P)$ is a scalar objective function that depends on $\boldsymbol{P}$ and $\boldsymbol{S}$ is a Lagrange multiplier. We are interested in computing the adjoints $\overline{\boldsymbol{A}}$, $\overline{\boldsymbol{B}}$, $\overline{\boldsymbol{Q}}$, and $\overline{\boldsymbol{R}}$. Taking partial derivatives on both sides of the Lagrangian and setting them to zero, we get

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{P}} = \frac{\partial \ell(\boldsymbol{P})}{\partial \boldsymbol{P}} + \tilde{\boldsymbol{A}}\boldsymbol{S} + \boldsymbol{S}\tilde{\boldsymbol{A}}^T \tag{4.47}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}} = \boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P} + \boldsymbol{Q} \tag{4.48}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{Q}} = \boldsymbol{S} \tag{4.49}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{R}} = \boldsymbol{K}\boldsymbol{S}\boldsymbol{K}^T \tag{4.50}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{A}} = \boldsymbol{P}\boldsymbol{S}^T + \boldsymbol{P}\boldsymbol{S} \tag{4.51}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{B}} = -\boldsymbol{P}\boldsymbol{S}\boldsymbol{K}^T - \boldsymbol{P}\boldsymbol{S}^T\boldsymbol{K}^T. \tag{4.52}$$

The Lagrange multiplier $\boldsymbol{S}$ satisfies the Lyapunov equation

$$\boldsymbol{0} = \frac{1}{2}\left(\overline{\boldsymbol{P}} + \overline{\boldsymbol{P}}^T\right) + \tilde{\boldsymbol{A}}\boldsymbol{S} + \boldsymbol{S}\tilde{\boldsymbol{A}}^T \tag{4.53}$$

where we have enforced symmetry of $\overline{\boldsymbol{P}}$ because $\boldsymbol{P}$ itself is symmetric – this ensures that $\boldsymbol{S}$ is also symmetric. The adjoints are given by:

$$\overline{\boldsymbol{A}} = 2\boldsymbol{P}\boldsymbol{S} \tag{4.54}$$

$$\overline{\boldsymbol{B}} = -2\boldsymbol{P}\boldsymbol{S}\boldsymbol{K} \tag{4.55}$$

$$\overline{\boldsymbol{Q}} = \boldsymbol{S} \tag{4.56}$$

$$\overline{\boldsymbol{R}} = \boldsymbol{K}\boldsymbol{S}\boldsymbol{K}^T. \tag{4.57}$$

4.5.2 Discrete time algebraic Riccati equation

The discrete time algebraic Riccati equation (DARE) is given by

$$\boldsymbol{A}^T\boldsymbol{P}\boldsymbol{A} - \boldsymbol{P} - (\boldsymbol{A}^T\boldsymbol{P}\boldsymbol{B})(\boldsymbol{R} + \boldsymbol{B}^T\boldsymbol{P}\boldsymbol{B})^{-1}(\boldsymbol{A}^T\boldsymbol{P}\boldsymbol{B})^T + \boldsymbol{Q} = \boldsymbol{0}. \tag{4.58}$$

Here, we derive $\dot{\boldsymbol{P}}$ and $\overline{\boldsymbol{P}}$ assuming that all conditions are satisfied that ensure the existence of a unique, symmetric solution $\boldsymbol{P}$.

Forward mode Taking the differential on both sides of Equation (4.43) and simplifying, we get:

$$\tilde{\boldsymbol{A}}^T d\boldsymbol{P}\, \tilde{\boldsymbol{A}} - d\boldsymbol{P} + \left(d\boldsymbol{Z} + d\boldsymbol{Z}^T + d\boldsymbol{Q} + \boldsymbol{K}^T\, d\boldsymbol{R}\, \boldsymbol{K}\right) = \boldsymbol{0} \tag{4.59}$$

where $\tilde{\boldsymbol{A}} = \boldsymbol{A} - \boldsymbol{B}\boldsymbol{K}$ and $\boldsymbol{K} = (\boldsymbol{R} + \boldsymbol{B}^T\boldsymbol{P}\boldsymbol{B})^{-1}\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{A}$, and $d\boldsymbol{Z} = \tilde{\boldsymbol{A}}^T\boldsymbol{P}\,(d\boldsymbol{A} - d\boldsymbol{B}\,\boldsymbol{K})$. Thus, in forward mode, we have the discrete time Lyapunov equation:

$$\tilde{\boldsymbol{A}}^T\dot{\boldsymbol{P}}\tilde{\boldsymbol{A}} - \dot{\boldsymbol{P}} + \left(\dot{\boldsymbol{Z}} + \dot{\boldsymbol{Z}}^T + \dot{\boldsymbol{Q}} + \boldsymbol{K}^T\dot{\boldsymbol{R}}\boldsymbol{K}\right) = \boldsymbol{0}. \tag{4.60}$$

where $\dot{\boldsymbol{Z}} = \tilde{\boldsymbol{A}}^T\boldsymbol{P}(\dot{\boldsymbol{A}} - \dot{\boldsymbol{B}}\boldsymbol{K})$.

Reverse mode We consider the Lagrangian

$$\begin{aligned}&\mathcal{L}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{P}, \boldsymbol{S}) = \\ &\quad f(\boldsymbol{P}) + \operatorname{tr}\left(\boldsymbol{S}^T(\boldsymbol{A}^T\boldsymbol{P}\boldsymbol{A} - \boldsymbol{P} - (\boldsymbol{A}^T\boldsymbol{P}\boldsymbol{B})(\boldsymbol{R} + \boldsymbol{B}^T\boldsymbol{P}\boldsymbol{B})^{-1}(\boldsymbol{A}^T\boldsymbol{P}\boldsymbol{B})^T + \boldsymbol{Q})\right).\end{aligned} \tag{4.61}$$

Taking partial derivatives on both sides of the Lagrangian and setting them to zero, we get

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{P}} = \frac{\partial f(\boldsymbol{P})}{\partial \boldsymbol{P}} + \tilde{\boldsymbol{A}}\boldsymbol{S}\tilde{\boldsymbol{A}}^T - \boldsymbol{S} \tag{4.62}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}} = \boldsymbol{A}^T\boldsymbol{P}\boldsymbol{A} - \boldsymbol{P} - (\boldsymbol{A}^T\boldsymbol{P}\boldsymbol{B})(\boldsymbol{R} + \boldsymbol{B}^T\boldsymbol{P}\boldsymbol{B})^{-1}(\boldsymbol{A}^T\boldsymbol{P}\boldsymbol{B})^T + \boldsymbol{Q} \tag{4.63}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{Q}} = \boldsymbol{S} \tag{4.64}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{R}} = \boldsymbol{K}\boldsymbol{S}\boldsymbol{K}^T \tag{4.65}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{A}} = \boldsymbol{P}\tilde{\boldsymbol{A}}\boldsymbol{S}^T + \boldsymbol{P}\tilde{\boldsymbol{A}}\boldsymbol{S} \tag{4.66}$$

$$0 = \frac{\partial \mathcal{L}}{\partial \boldsymbol{B}} = -\boldsymbol{P}\tilde{\boldsymbol{A}}\boldsymbol{S}\boldsymbol{K}^T - \boldsymbol{P}\tilde{\boldsymbol{A}}\boldsymbol{S}^T\boldsymbol{K}^T \tag{4.67}$$

The Lagrange multiplier $\boldsymbol{S}$ satisfies the Lyapunov equation

$$\boldsymbol{0} = \frac{1}{2}(\overline{\boldsymbol{P}} + \overline{\boldsymbol{P}}^T) + \tilde{\boldsymbol{A}}\boldsymbol{S}\tilde{\boldsymbol{A}}^T - \boldsymbol{S} \tag{4.68}$$

where we have enforced the symmetry of $\overline{\boldsymbol{P}}$ because $\boldsymbol{P}$ is symmetric. This ensures that $\boldsymbol{S}$ is also symmetric and the adjoints are given by:

$$\overline{\boldsymbol{A}} = 2\boldsymbol{P}\tilde{\boldsymbol{A}}\boldsymbol{S} \tag{4.69}$$

$$\overline{\boldsymbol{B}} = -2\boldsymbol{P}\tilde{\boldsymbol{A}}\boldsymbol{S}\boldsymbol{K}^T \tag{4.70}$$

$$\overline{\boldsymbol{Q}} = \boldsymbol{S} \tag{4.71}$$

$$\overline{\boldsymbol{R}} = \boldsymbol{K}\boldsymbol{S}\boldsymbol{K}^T. \tag{4.72}$$

4.6 Example application: inverse LQR

To show a concrete application of automatic differentiation through these algebraic matrix equations, we consider a discrete time linear system that evolves according to the equations

$$\boldsymbol{x}_{t+1} = \boldsymbol{A}\boldsymbol{x}_t + \boldsymbol{B}\boldsymbol{u}_t, \quad \boldsymbol{x}_0 = \boldsymbol{x}_{\text{init}} \tag{4.73}$$

where $\boldsymbol{x}_t$ is the state of the system, $\boldsymbol{u}_t$ is some input that enters the system through some matrix $\boldsymbol{B}$, and $\boldsymbol{A}$ is the state transition matrix. The infinite-horizon, discrete-time LQR problem involves finding the optimal inputs $\{\boldsymbol{u}_0, \boldsymbol{u}_1, \ldots\}$ that minimise [4]

$$J[\boldsymbol{u}_0, \boldsymbol{u}_1, \cdots] = \sum_{t=0}^{\infty} \boldsymbol{x}_t^T\boldsymbol{Q}\boldsymbol{x}_t + \boldsymbol{u}_t^T\boldsymbol{R}\boldsymbol{u}_t \tag{4.74}$$

for some symmetric positive semi-definite matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$. It is well-known that the optimal solution is linear state-feedback:

$$\boldsymbol{u}_t = -\boldsymbol{K}\boldsymbol{x}_t, \tag{4.75}$$

where $\boldsymbol{K} = (\boldsymbol{R} + \boldsymbol{B}^T\boldsymbol{P}\boldsymbol{B})^{-1}\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{A}$ and $\boldsymbol{P}$ satisfies a DARE (c.f. Equation (4.58)):

$$\boldsymbol{A}^T\boldsymbol{P} + \boldsymbol{P}\boldsymbol{A} - \boldsymbol{P}\boldsymbol{B}\boldsymbol{R}^{-1}\boldsymbol{B}^T\boldsymbol{P} + \boldsymbol{Q} = \boldsymbol{0}. \tag{4.76}$$

We consider the following inverse LQR problem: can we infer $\boldsymbol{Q}$, i.e. the way state deviations from zero are penalized in various state space directions, given sample state trajectories generated by the optimally controlled system, and given knowledge of $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{R}$? More formally, given K trajectories of the system under optimal LQR control sampled at T time points:

$$\boldsymbol{x}_0^{(k)}, \boldsymbol{x}_1^{(k)}, \cdots \boldsymbol{x}_{T-1}^{(k)} \quad \text{for} \quad k = 1, \cdots, K, \tag{4.77}$$

can we determine the matrix $\boldsymbol{Q}$ used in the LQR algorithm's objective function (Equation (4.74))? Assuming that this problem is well-posed (i.e., $\boldsymbol{Q}$ is identifiable; 215), we can solve this problem by differentiating through DARE (Section 4.5.2). More specifically, we start with an initial guess of $\boldsymbol{Q}$, which we denote as $\hat{\boldsymbol{Q}}$. Next, we solve the LQR problem and find the corresponding optimal inputs $\hat{\boldsymbol{u}}_t$ and trajectories $\hat{\boldsymbol{x}}_t$. We then minimize the objective

$$\ell(\hat{\boldsymbol{Q}}) = \frac{1}{KT}\sum_{k=1}^{K}\sum_{t=0}^{T-1}(\boldsymbol{x}_t^{(k)} - \hat{\boldsymbol{x}}_t^{(k)})^2. \tag{4.78}$$

with respect to $\hat{\boldsymbol{Q}}$. To illustrate our solution, we consider the following example system:

$$\begin{aligned} \boldsymbol{A} &= \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \qquad \boldsymbol{B} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\ \boldsymbol{Q} &= \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \qquad \boldsymbol{R} = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.3 \end{pmatrix}. \end{aligned} \tag{4.79}$$

We created a synthetic data set by sampling $\boldsymbol{K} = 30$ initial states $\boldsymbol{x}_{\text{init}}$ from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and for each sample, we simulated the activity trajectories under optimal control for $T = 30$ steps. We then optimized $\ell(\hat{\boldsymbol{Q}})$ for the synthetic data set using L-BFGS [102]. After 7 iterations, we were able to recover the true $\boldsymbol{Q}$ to a good degree of accuracy (Figure 4.1). Code for this example is available at https://github.com/tachukao/autodiff-inverse-lqr.

4.7 Conclusion

In this note, we derived the forward and reverse mode derivatives for the solutions to the discrete time and continuous time variants of the Sylvester, Lyapunov, and Riccati equations (summarized in Tables 4.1 and 4.2). These equations are widely-used in control theory and other branches of applied mathematics. We demonstrate the usefulness of these derivatives on an inverse LQR

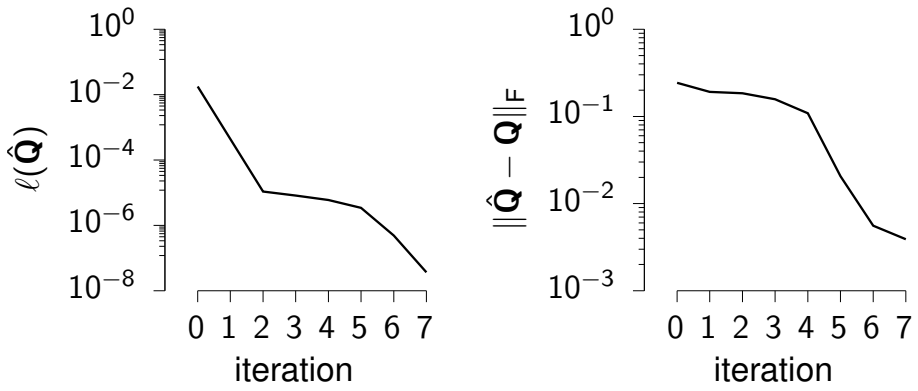


Figure 4.1: Evolution of $\ell(\hat{\boldsymbol{Q}})$ (left) and $\|\hat{\boldsymbol{Q}}\|$ (right) duing optimization (see text).

Equation	**Tangents**
CONTINUOUS TIME SYLVESTER Equation (4.1)	
$\boldsymbol{AP+PB+C=0}$	$\boldsymbol{A\dot{P}+\dot{P}B+\left(\dot{A}P+P\dot{B}+\dot{C}\right)=0}$
DISCRETE TIME SYLVESTER Equation (4.12)	
$\boldsymbol{APB-P+C=0}$	$\boldsymbol{A\dot{P}B-\dot{P}+\left(\dot{A}PB+AP\dot{B}+\dot{C}\right)=0}$
CONTINUOUS TIME LYAPUNOV Equation (4.23)	
$\boldsymbol{AP+PA^T+C=0}$	$\boldsymbol{A\dot{P}+\dot{P}A^T+\left(\dot{A}P+P\dot{A}^T+\dot{C}\right)=0}$
DISCRETE TIME LYAPUNOV Equation (4.33)	
$\boldsymbol{APA^T-P+C=0}$	$\boldsymbol{A\dot{P}A^T-\dot{P}+\left(\dot{A}PA^T+AP\dot{A}^T+\dot{C}\right)=0}$
CONTINUOUS TIME ALGEBRAIC RICCATI Equation (4.43)	
$\boldsymbol{A^TP+PA-PBR^{-1}B^TP+Q=0}$	$\boldsymbol{\dot{P}\tilde{A}+\tilde{A}^T\dot{P}+\left(\dot{A}^TP+P\dot{A}-P\dot{B}K-K^T\dot{B}^TP+\dot{Q}+K^T\dot{R}K\right)=0}$
DISCRETE TIME ALGEBRIAC RICCATI Equation (4.58)	
$\boldsymbol{A^TPA-P-(A^TPB)(R+B^TPB)^{-1}(B^TPA)+Q=0}$	$\boldsymbol{\tilde{A}^T\dot{P}\tilde{A}-\dot{P}+\left(\dot{A}^TP\tilde{A}+\tilde{A}^TP\dot{A}-K^T\dot{B}^TP\tilde{A}-\tilde{A}^TP\dot{B}K+\dot{Q}+K^T\dot{R}K\right)=0}$

Table 4.1: Summary of forward-mode derivatives

problem, where parameters of the optimal control problem are inferred from observations of state trajectories by differentiating through the discrete time Riccati equation (Section 4.5.2). These derivatives (with the exception of the discrete time Sylvester equation, of more limited use), have been implemented and numerically tested in Owl, a numerical library written in OCaml with a full-featured automatic differentiation module.

Equation	Adjoints
continuous time Sylvester Equation (4.1)	
$AP+PB+C=0$	$0=A^TS+SB^T+\overline{P}$ $\overline{A}=SP^T \quad \overline{B}=P^TS \quad \overline{C}=S$
discrete time Sylvester Equation (4.12)	
$APB-P+C=0$	$0=A^TSB^T-S+\overline{P}$ $\overline{A}=SB^TP^T \quad \overline{B}=P^TA^TS \quad \overline{C}=S$
continuous time Lyapunov Equation (4.23)	
$AP+PA^T+C=0$	$0=\overline{P}+A^TS+SA$ $\overline{A}=SP^T+S^TP \quad \overline{Q}=S$
discrete time Lyapunov Equation (4.33)	
$APA^T-P+C=0$	$0=\overline{P}+A^TSA-S$ $\overline{A}=SAP^T+S^TAP \quad \overline{Q}=S.$
continuous time algebraic Riccati Equation (4.43)	
$A^TP+PA-PBR^{-1}B^TP+Q=0$	$0=\frac{1}{2}(\overline{P}+\overline{P}^T)+\tilde{A}S+S\tilde{A}^T$ $\overline{A}=2PS \quad \overline{B}=-2PSK$ $\overline{Q}=S \quad \overline{R}=KSK^T.$
discrete time algebraic Riccati Equation (4.58)	
A^TPA-P $-(A^TPB)(R+B^TPB)^{-1}(B^TPA)+Q=0$	$0=\frac{1}{2}(\overline{P}+\overline{P}^T)+\tilde{A}S\tilde{A}^T-S$ $\overline{A}=2P\tilde{A}S \quad \overline{B}=-2P\tilde{A}SK^T$ $\overline{Q}=S \quad \overline{R}=KSK^T$

Table 4.2: Summary of reverse-mode derivatives

Chapter 5

Manifold GPLVMs for discovering non-Euclidean latent structure in neural data

Summary

This chapter presents the following article:

> Manifold GPLVMs for discovering non-Euclidean latent structure in neural data. Kristopher T. Jensen, **Ta-Chu Kao**, Marco Tripodi, and Guillaume Hennequin (2020). *Advances in Neural Information Processing Systems*, 33, 22580–22592.

A common problem in neuroscience is to elucidate the collective neural representations of behaviorally important variables such as head direction, spatial location, upcoming movements, or mental spatial transformations. Often, these latent variables are internal constructs not directly accessible to the experimenter. Here, we propose a new probabilistic latent variable model to simultaneously identify the latent state and the way each neuron contributes to its representation in an unsupervised way. In contrast to previous models which assume Euclidean latent spaces, we embrace the fact that latent states often belong to symmetric manifolds such as spheres, tori, or rotation groups of various dimensions. We therefore propose the manifold Gaussian process latent variable model (mGPLVM), where neural responses arise from (i) a shared latent variable living on a specific manifold, and (ii) a set of non-parametric tuning curves determining how each neuron contributes to the representation. Cross-validated comparisons of models with different topologies can be used to distinguish between candidate manifolds, and variational

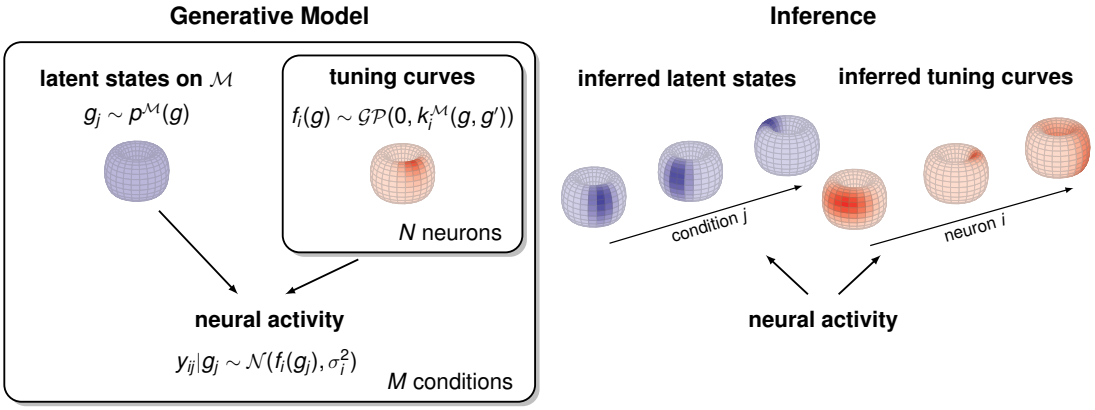


Figure 5.1: **Schematic illustration of the manifold Gaussian process latent variable model (mGPLVM).** In the generative model (left), neural activity arises from (i) M latent states $\{g_j\}$ on a manifold $\mathcal{M}$, each corresponding to a different condition j (e.g. time or stimulus), and (ii) the tuning curves of N neurons, modelled as Gaussian processes and sharing the same latent states $\{g_j\}$ as inputs. Using variational inference, mGPLVM jointly infers the global latent states and the tuning curve of each neuron on the manifold (right).

inference enables quantification of uncertainty. We demonstrate the validity of the approach on several synthetic datasets, as well as on calcium recordings from the ellipsoid body of *Drosophila melanogaster* and extracellular recordings from the mouse anterodorsal thalamic nucleus. These circuits are both known to encode head direction, and mGPLVM correctly recovers the ring topology expected from neural populations representing a single angular variable.

5.1 Introduction

The brain uses large neural populations to represent low-dimensional quantities of behavioural relevance such as location in physical or mental spaces, orientation of the body, or motor plans. It is therefore common to project neural data into smaller latent spaces as a first step towards linking neural activity to behaviour [27]. This can be done using a variety of linear methods such as PCA or factor analysis [26], or non-linear dimensionality reduction techniques such as tSNE [106]. Many of these methods are explicitly probabilistic, with notable examples including GPFA [213] and LFADS [131]. However, all these models project data into Euclidean latent spaces, thus failing to capture the inherent non-Euclidean nature of variables such as head direction or rotational motor plans [17, 45, 162, 204].

Most models in neuroscience justifiably assume that neurons are smoothly tuned [172]. As an example, a population of neurons representing an angular variable θ would respond similarly

to some θ and to $\theta + \epsilon$ (for small ϵ). While it is straightforward to model such smoothness by introducing smooth priors for response functions defined over $\mathbb{R}$, the activity of neurons modelled this way would exhibit a spurious discontinuity as the latent angle changes from 2π to $0 + \epsilon$. We see that appropriately modelling smooth neuronal representations requires keeping the latent variables of interest on their natural manifold (here, the circle), instead of an ad-hoc Euclidean space. While periodic kernels have commonly been used to address such problems in GP regression [107], topological structure has not been incorporated into GP-based latent variable models due to the difficulty of doing inference in such spaces.

Here, we build on recent advances in non-Euclidean variational inference [41] to develop the manifold Gaussian process latent variable model (mGPLVM), an extension of the GPLVM framework [96, 182, 209, 210] to non-Euclidean latent spaces including tori, spheres and $SO(3)$ (Figure 5.1). mGPLVM jointly learns the fluctuations of an underlying latent variable g *and* a probabilistic "tuning curve" $p(f_i|g)$ for each neuron i. The model therefore provides a fully unsupervised way of querying how the brain represents its surroundings and a readout of the relevant latent quantities. Importantly, the probabilistic nature of the model enables principled model selection between candidate manifolds. We provide a framework for scalable inference and validate the model on both synthetic and experimental datasets.

5.2 Manifold Gaussian process latent variable model

The main contribution of this paper is mGPLVM, a Gaussian process latent variable model [182, 209] defined for non-Euclidean latent spaces. We first present the generative model (Section 5.2.1), then explain how we perform approximate inference using reparameterizations on Lie groups (41; Section 5.2.2). Lie groups include Euclidean vector spaces $\mathbb{R}^n$ as well as other manifolds of interests to neuroscience such as tori T^n [17, 152] and the special orthogonal group $SO(3)$ (45, 204; extensions to non-Lie groups are discussed in Section 5.5.5). We then provide specific forms for variational densities and kernels on tori, spheres, and $SO(3)$ (Section 5.2.3). Finally we validate the method on both synthetic data (Section 5.3.1), calcium recordings from the fruit fly head direction system (Section 5.3.2), and extracellular recordings from the mouse anterodorsal thalamic nucleus (Section 5.5.1).

5.2.1 Generative model

We use x_{ij} to denote the individual elements of a matrix $\boldsymbol{X}$. Let $\boldsymbol{Y} \in \mathbb{R}^{N \times M}$ be the activity of N neurons recorded in each of M conditions. Examples of "conditions" include time within a trial, stimulus identity, or motor output. We assume that all neuronal responses collectively encode a shared, condition-specific latent variable $g_j \in \mathcal{M}$, where $\mathcal{M}$ is some manifold. We further assume that each neuron i is tuned to the latent state g with a "tuning curve" $f_i(g)$, describing its average response conditioned on g. Rather than assuming a specific parametric form for these tuning curves, we place a Gaussian process prior on $f_i(\cdot)$ to capture the heterogeneity widely observed in biological systems [23, 68]. The model is depicted in Figure 5.1 and can be formally described as:

$$g_j \sim p^{\mathcal{M}}(g) \qquad \text{(prior over latents)} \tag{5.1}$$

$$f_i \sim \mathcal{GP}(0, k_i^{\mathcal{M}}(\cdot,\cdot)) \qquad \text{(prior over tuning curves)} \tag{5.2}$$

$$y_{ij}|g_j \sim \mathcal{N}(f_i(g_j), \sigma_i^2) \qquad \text{(noise model)} \tag{5.3}$$

In Equation (5.1), we use a uniform prior $p^{\mathcal{M}}(g)$ inversely proportional to the volume of the manifold for bounded manifolds (Section 5.5.2), and a Gaussian prior on Euclidean spaces to set a basic lengthscale. In Equation (5.2), $k_i^{\mathcal{M}}(\cdot,\cdot) : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ is a covariance function defined on manifold $\mathcal{M}$ – manifold-specific details are discussed in Section 5.2.3. In the special case where $\mathcal{M}$ is a Euclidean space, this model is equivalent to the standard Bayesian GPLVM [182]. While Equation (5.3) assumes independent noise across neurons, noise correlations can also be introduced as in [209] and Poisson noise as in [210].

This probabilistic model can be fitted by maximizing the log marginal likelihood

$$\log p(\boldsymbol{Y}) = \log \int p(\boldsymbol{Y}|\{f_i\},\{g_j\})\, p(\{f_i\})\; p^{\mathcal{M}}(\{g_j\})\; d\{f_i\} d\{g_j\}. \tag{5.4}$$

Following optimization, we can query both the posterior over latent states $p(\{g_j\}|\boldsymbol{Y})$ and the posterior predictive distribution $p(\boldsymbol{Y}^\star|\mathcal{G}^\star, \boldsymbol{Y})$ at a set of query states $\mathcal{G}^\star$. While it is possible to marginalise out f_i when the states $\{g_j\}$ are known, further marginalising out $\{g_j\}$ is intractable and maximizing Equation (5.4) requires approximate inference.

5.2.2 Learning and inference

To maximize $\log p(\boldsymbol{Y})$ in Equation (5.4), we use variational inference as previously proposed for GPLVMs [182]. The true posterior over the latent states, $p(\{g_j\}|\mathbf{Y})$, is approximated by a variational distribution $Q_\theta(\{g_j\})$ with parameters θ that are optimized to minimize the KL

divergence between $Q_\theta(\{g_j\})$ and $p(\{g_j\}|\boldsymbol{Y})$. This is equivalent to maximizing the evidence lower bound (ELBO) on the log marginal likelihood:

$$\mathcal{L}(\theta) = H(Q_\theta) + \mathbb{E}_{Q_\theta}[\log p^{\mathcal{M}}(\{g_j\})] + \mathbb{E}_{Q_\theta}[\log p(\boldsymbol{Y}|\{g_j\})]. \tag{5.5}$$

Here, $\mathbb{E}_{Q_\theta}[\cdot]$ indicates averaging over the variational distribution and $H(Q_\theta)$ is its entropy. For simplicity, and because our model does not specify *a priori* statistical dependencies between the individual elements of $\{g_j\}$, we choose a variational distribution Q_θ that factorizes over conditions:

$$Q_\theta(\{g_j\}) = \prod_{j=1}^{M} q_{\theta_j}(g_j). \tag{5.6}$$

In the Euclidean case, the entropy and expectation terms in Equation (5.5) can be calculated analytically for some kernels [182], and otherwise using the reparameterization trick [91, 147]. Briefly, the reparameterization trick involves first sampling from a fixed, easy-to-sample distribution (e.g. a normal distribution with zero mean and unit variance), and applying a series of differentiable transformations to obtain samples from Q_θ. We can then use these samples to estimate the entropy term and expectations in Equation (5.5).

For non-Euclidean manifolds, inference in mGPLVMs poses two major problems. Firstly, we can no longer calculate the ELBO analytically nor evaluate it using the standard reparameterization trick. Secondly, evaluating the Gaussian process log marginal likelihood $\log p(\boldsymbol{Y}|\{g_j\})$ exactly becomes computationally too expensive for large datasets. We address these issues in the following.

Reparameterizing distributions on Lie groups

To estimate and optimize the ELBO in Equation (5.5) when Q_θ is defined on a non-Euclidean manifold, we use Falorsi et al.'s ReLie framework, an extension of the standard reparameterization trick to variational distributions defined on Lie groups.

Sampling from $\boldsymbol{Q_\theta}$ Since we assume that Q_θ factorizes (Equation (5.6)), sampling from Q_θ is performed by independently sampling from each q_{θ_j}. We start from a differentiable base distribution $r_{\theta_j}(\boldsymbol{x})$ in $\mathbb{R}^n$. Note that $\mathbb{R}^n$ is isomorphic to the tangent space at the identity element of the group G, known as the Lie algebra. We can thus define a 'capitalized' exponential map $\mathrm{Exp}_G : \mathbb{R}^n \to G$, which maps elements of $\mathbb{R}^n$ to elements in G (170; Section 5.5.3). Importantly, Exp_G maps a distribution centered at zero in $\mathbb{R}^n$ to a distribution $\tilde{q}_{\theta_j}$ in the group centered at the identity element. To obtain samples from a distribution q_{θ_j} centered at an arbitrary g_j^μ in

the group, we can simply apply the group multiplication with g_j^μ to samples from $\tilde{q}_{\theta_j}$. Therefore, obtaining a sample g_j from q_{θ_j} involves the following steps: (i) sample from $r_{\theta_j}(\boldsymbol{x})$, (ii) apply Exp_G to obtain a sample $\tilde{g}_j$ from $\tilde{q}_{\theta_j}$, and (iii) apply the group multiplication $g_j = g_j^\mu \tilde{g}_j$.

Estimating the entropy $\boldsymbol{H}(\boldsymbol{Q}_\theta)$ Since $H(q_{\theta_j}) = H(\tilde{q}_{\theta_j})$ [41], we use K independent Monte Carlo samples from $\tilde{Q}_\theta(\cdot) = \prod_{j=1}^M \tilde{q}_{\theta_j}(\cdot)$ to calculate

$$H(Q_\theta) \approx -\frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{M} \log \tilde{q}_{\theta_j}(\tilde{g}_{jk}), \tag{5.7}$$

where $\tilde{g}_{jk} = \mathrm{Exp}_G \boldsymbol{x}_{jk}$ and $\{\boldsymbol{x}_{jk} \sim r_{\theta_j}(\boldsymbol{x})\}_{k=1}^K$.

Evaluating the density $\tilde{\boldsymbol{q}}_\theta$ To evaluate $\log \tilde{q}_{\theta_j}(\mathrm{Exp}_G \boldsymbol{x}_{jk})$, we use the result from Falorsi et al. [41] that

$$\tilde{q}_\theta(\tilde{g}) = \sum_{\boldsymbol{x}\in\mathbb{R}^n \,:\, \mathrm{Exp}_G(\boldsymbol{x})=\tilde{g}} r_\theta(\boldsymbol{x})|\boldsymbol{J}(\boldsymbol{x})|^{-1} \tag{5.8}$$

where $\boldsymbol{J}(\boldsymbol{x})$ is the Jacobian of Exp_G at $\boldsymbol{x}$. Thus, $\tilde{q}_\theta(\tilde{g})$ is the sum of the Jacobian-weighted densities $r_\theta(\boldsymbol{x})$ in $\mathbb{R}^n$ at *all* those points that are mapped to $\tilde{g}$ through Exp_G This is an infinite but converging sum, and following Falorsi et al. [41] we approximate it by its first few dominant terms (Section 5.5.10).

Note that $\mathrm{Exp}_G(\cdot)$ and the group multiplication by g^μ are both differentiable operations. Therefore, as long as we choose a differentiable base distribution $r_\theta(\boldsymbol{x})$, we can perform end-to-end optimization of the ELBO. In this work we choose the reference distribution to be a multivariate normal $r_{\theta_j}(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; 0, \boldsymbol{\Sigma}_j)$ for each q_{θ_j}. We variationally optimize both $\{\boldsymbol{\Sigma}_j\}$ and the mean parameters $\{g_j^\mu\}$ for all j, and together these define the variational distribution.

Sparse GP approximation

To efficiently evaluate the $\mathbb{E}_{Q_\theta}[\log p(\boldsymbol{Y}|\{g_j\})]$ term in the ELBO for large datasets, we use the variational sparse GP approximation [181] which has previously been applied to Euclidean GPLVMs [182]. Specifically, we introduce a set of m inducing points $\mathcal{Z}_i$ for each neuron i, and use a lower bound on the GP log marginal likelihood:

$$\log p(\boldsymbol{y}_i|\{g_j\}) \geq \underbrace{-\frac{1}{2}\boldsymbol{y}_i^T(\boldsymbol{Q}_i + \sigma_i^2\boldsymbol{I})^{-1}\boldsymbol{y}_i - \frac{1}{2}\log|\boldsymbol{Q}_i + \sigma_i^2\boldsymbol{I}| - \frac{1}{2\sigma^2}\mathrm{Tr}(\boldsymbol{K}_i - \boldsymbol{Q}_i) + \text{const.}}_{\log \tilde{p}(\boldsymbol{y}_i|\{g_j\})} \tag{5.9}$$

$$\text{with } \boldsymbol{Q}_i = \boldsymbol{K}_{\{g_j\}\mathcal{Z}_i}\boldsymbol{K}_{\mathcal{Z}_i\mathcal{Z}_i}^{-1}\boldsymbol{K}_{\mathcal{Z}_i\{g_j\}} \tag{5.10}$$

where $\boldsymbol{K}_{\mathcal{AB}}$ denotes the Gram matrix associated with any two input sets $\mathcal{A}$ and $\mathcal{B}$. Note that the latents $\{g_j\}$ are shared across all neurons. In this work we optimize the inducing points on G directly, but they could equivalently be optimized in $\mathbb{R}^n$ and projected onto G via Exp_G.

Using the sparse GP framework, the cost of computing the GP likelihood reduces to $\mathcal{O}(Mm^2)$ for each neuron and Monte Carlo sample. This leads to an overall complexity of $\mathcal{O}(KNMm^2)$ for approximating $\mathbb{E}_{Q_\theta}[\log p(\boldsymbol{Y}|\{g_j\})]$ with K Monte Carlo samples, N neurons, M conditions and m inducing points (see Section 5.5.10 for further details on complexity and implementation).

Optimization

We are now equipped to optimize the ELBO defined in Equation (5.5) using Monte Carlo samples drawn from a variational distribution Q_θ defined on a Lie group G. To train the model, we use Adam [90] to perform stochastic gradient descent on the following loss function:

$$\mathcal{L}(\theta) = \frac{1}{K}\sum_{k=1}^{K}\left[\sum_{j=1}^{M}\left(\log p^{\mathcal{M}}(g_{jk}) - \log \tilde{q}_{\theta_j}(\tilde{g}_{jk})\right) - \sum_{i}^{N}\log \tilde{p}(\boldsymbol{y}_i|\{g_{jk}\})\right] \tag{5.11}$$

where a set of K Monte-Carlo samples $\{\tilde{g}_{jk}\}_{k=1}^{K}$ is drawn at each iteration from $\{\tilde{q}_{\theta_j}\}$ as described in Section 5.2.2. In Equation (5.11), $g_{jk} = g_j^{\mu}\tilde{g}_{jk}$, where g_j^{μ} is a group element that is optimized together with all other model parameters. Finally, $\log \tilde{p}(\boldsymbol{y}_i|\{g_j\})$ is the lower bound defined in Equation (5.9) and $p^{\mathcal{M}}(g_{jk})$ is the prior described in Section 5.2.1. The inner sums run over conditions j and neurons i.

Posterior over tuning curves

We approximate the posterior predictive distribution over tuning curves by sampling from the (approximate) posterior over latents. Specifically, for a given neuron i and a set of query states $\mathcal{G}^\star$, the posterior predictive over $\boldsymbol{f}_i^\star$ is approximated by:

$$p(\boldsymbol{f}_i^\star|\boldsymbol{Y}, \mathcal{G}^\star) = \frac{1}{K}\sum_{k=1}^{K} p(\boldsymbol{f}_i^\star|\mathcal{G}^\star, \{\mathcal{G}_k, \boldsymbol{Y}\}) \tag{5.12}$$

where each $\mathcal{G}_k$ is a set of M latent states (one for each condition in $\boldsymbol{Y}$) independently drawn from the variational posterior $Q_\theta(\cdot)$. In Equation (5.12), each term in the sum is a standard Gaussian process posterior [143], which we approximate as described above (Section 5.2.2; Section 5.5.6; 181).

5.2.3 Applying mGPLVM to tori, spheres and SO(3)

At this stage, we have yet to define the manifold-specific GP kernels $k^{\mathcal{M}}$ described in Section 5.2.1. These kernels ought to capture the topology of the latent space and express our prior assumptions that the neuronal tuning curves, defined on the manifold, have certain properties such as smoothness. Here we take inspiration from the common squared exponential covariance function defined over Euclidean spaces and introduce analogous kernels on tori, spheres, and $SO(3)$. This leads to the following general form:

$$k^{\mathcal{M}}(g, g') = \alpha^2 \exp\left(-\frac{d_{\mathcal{M}}(g, g')}{2\ell^2}\right) \qquad g, g' \in \mathcal{M} \tag{5.13}$$

where α^2 is a variance parameter, ℓ is a characteristic lengthscale, and $d_{\mathcal{M}}(g, g')$ is a manifold-specific distance function. While squared geodesic distances might be intuitive choices for $d(\cdot, \cdot)$ in Equation (5.13), they result in positive semi-definite (PSD) kernels only for Euclidean latent spaces [43, 80]. Therefore, we build distance functions that automatically lead to valid covariance functions by observing that (i) dot product kernels are PSD, and (ii) the exponential of a PSD kernel is also PSD. Specifically, we use the following manifold-specific dot product-based distances:

$$d_{R^n}(g, g') = ||g - g'||_2^2 \qquad g \in \mathbb{R}^n \tag{5.14}$$

$$d_{S^n}(g, g') = 2(1 - g \cdot g') \qquad g \in \{\boldsymbol{x} \in \mathbb{R}^{n+1};\ \|\boldsymbol{x}\| = 1\} \tag{5.15}$$

$$d_{T^n}(g, g') = 2\textstyle\sum_k (1 - g_k \cdot g'_k) \qquad g \in \{(g_1, \cdots, g_n);\ \forall k:\ g_k \in \mathbb{R}^2, \|g_k\| = 1\} \tag{5.16}$$

$$d_{SO(3)}(g, g') = 4\left[1 - (g \cdot g')^2\right] \qquad g \in \{\boldsymbol{x} \in \mathbb{R}^4;\ \|\boldsymbol{x}\| = 1\} \tag{5.17}$$

where we have slightly abused notation by directly using "g" to denote a convenient parameterisation of the group elements which we define on the right of each equation. To build intuition, we note that the distance metric on the torus gives rise to a multivariate von Mises function; the distance metric on the sphere leads to an analogous von Mises Fisher function; and the distance metric on $SO(3)$ is $2(1 - \cos\varphi_{\text{rot}})$ where φ_{rot} is the angle of rotation required to transform g into g'. Notably, all these distance functions reduce to the Euclidean squared exponential kernel in the small angle limit. Laplacian [43] and Matérn [12] kernels have previously been proposed for modelling data on Riemannian manifolds, and these can also be incorporated in mGPLVM.

Finally, we provide expressions for the variational densities (Equation (5.8)) defined on tori, S^3

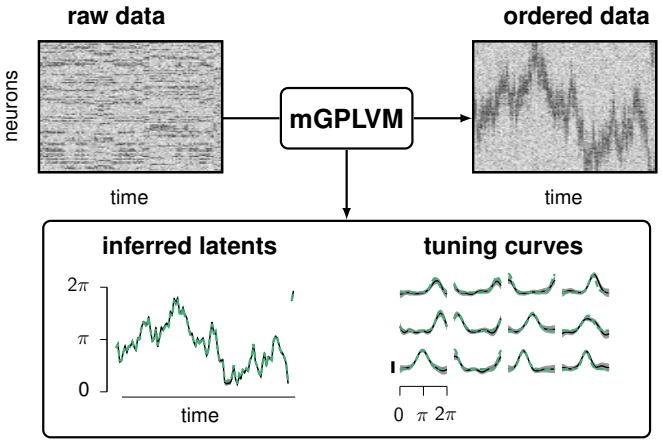


Figure 5.2: **Applying mGPLVM to synthetic data on the ring T^1. Top left**: neural activity of 100 neurons at 100 different conditions (here, time bins). **Bottom**: timecourse of the latent states (left) and tuning curves for 12 representative neurons (right). Green: ground truth; Black: posterior mean; Grey shaded regions: ±2 posterior s.t.d. **Top right**: data replotted from the top left panel, with neurons reordered according to their preferred angles as determined by the inferred tuning curves.

and $SO(3)$:

$$\tilde{q}_\theta(\mathrm{Exp}_{T^n}\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^n} r_\theta(\boldsymbol{x}+2\pi\boldsymbol{k}), \tag{5.18}$$

$$\tilde{q}_\theta(\mathrm{Exp}_{SO(3)}\boldsymbol{x}) = \sum_{k\in\mathbb{Z}} \left[r_\theta(\boldsymbol{x}+\pi k\hat{\boldsymbol{x}})\, \frac{2\|\boldsymbol{x}+\pi k\hat{\boldsymbol{x}}\|^2}{1-\cos\left(2\|\boldsymbol{x}+\pi k\hat{\boldsymbol{x}}\|\right)} \right], \tag{5.19}$$

$$\tilde{q}_\theta(\mathrm{Exp}_{S^3}\boldsymbol{x}) = \sum_{k\in\mathbb{Z}} \left[r_\theta(\boldsymbol{x}+2\pi k\hat{\boldsymbol{x}})\, \frac{2\|\boldsymbol{x}+2\pi k\hat{\boldsymbol{x}}\|^2}{1-\cos\left(2\|\boldsymbol{x}+2\pi k\hat{\boldsymbol{x}}\|\right)} \right], \tag{5.20}$$

where $\hat{\boldsymbol{x}} = \boldsymbol{x}/\|\boldsymbol{x}\|$. Further details and the corresponding exponential maps are given in Section 5.5.3. Since spheres that are not S^1 or S^3 are not Lie groups, ReLie does not provide a general framework for mGPLVM on these manifolds which we therefore treat separately in Section 5.5.5.

5.3 Experiments and results

In this section, we start by demonstrating the ability of mGPLVM to correctly infer latent states and tuning curves in non-Euclidean spaces using synthetic data generated on T^1, T^2 and $SO(3)$. We also verify that cross-validated model comparison correctly recovers the topology of the underlying latent space, suggesting that mGPLVM can be used for model selection given a set of candidate manifolds. Finally, we apply mGPLVM to a biological dataset to show that it is robust to the noise and heterogeneity characteristic of experimental recordings.

5.3.1 Synthetic data

To generate synthetic data $\mathbf{Y}$, we specify a target manifold $\mathcal{M}$, draw a set of M latent states $\{g_j\}$ on $\mathcal{M}$, and assign a tuning curve to each neuron i of the form

$$f_i(g) = a_i^2 \exp\left(-\frac{d_{\text{geo}}^2(g, g_i^{\text{pref}})}{2b_i^2}\right) + c_i, \tag{5.21}$$

$$y_{ij}|g_j \sim \mathcal{N}(f_i(g_j), \sigma_i^2) \tag{5.22}$$

with random parameters a_i, b_i and c_i. Thus, the activity of each neuron is a noisy bell-shaped function of the geodesic distance on $\mathcal{M}$ between the momentary latent state g_j and the neuron's preferred state g_i^{pref} (sampled uniformly). While this choice of tuning curves is inspired by the common 'Gaussian bump' model of neural tuning, we emphasize that the non-parametric prior over f_i in mGPLVM can discover any smooth tuning curve on the manifold, not just Gaussian bumps. For computational simplicity, here we constrain the mGPLVM parameters α_i, ℓ_i and σ_i to be identical across neurons. Note that we can only recover the latent space up to symmetries which preserve pairwise distances. In all figures, we have therefore aligned model predictions and ground truth for ease of visualization (Section 5.5.7).

We first generated data on the ring (T^1, Figure 5.2, top left), letting the true latent state be a continuous random walk across conditions for ease of visualization. We then fitted T^1-mGPLVM to the data and found that it correctly discovered the true latent states g as well as the ground truth tuning curves (Figure 5.2, bottom right). Reordering the neurons according to their preferred angles further exposed the population encoding of the angle (Figure 5.2, top right).

Next, we expanded the latent space to two dimensions with data now populating a 2-torus (T^2). Despite the non-trivial topology of this space, T^2-mGPLVM provided accurate inference of both latent states (Figure 5.3a) and tuning curves (Figure 5.3b). To show that mGPLVM can be used to distinguish between candidate topologies, we compared T^2-mGPLVM to a standard Euclidean GPLVM in $\mathbb{R}^2$ on the basis of both cross-validated prediction errors and importance-weighted marginal likelihood estimates [14]. We simulated 10 different toroidal datasets; for each, we used half the conditions to fit the GP hyperparameters, and half the neurons to predict the latent states for the conditions not used to fit the GP parameters. Finally, we used the inferred GP parameters and latent states to predict the activity of the held-out neurons at the held-out conditions. As expected, the predictions of the toroidal model outperformed those of the standard Euclidean GPLVM which cannot capture the periodic boundary conditions of the torus (Figure 5.3c).

Beyond toroidal spaces, $SO(3)$ is of particular interest for the study of neural systems encoding 'yaw, pitch and roll' in a variety of 3D rotational contexts [45, 167, 204]. We therefore

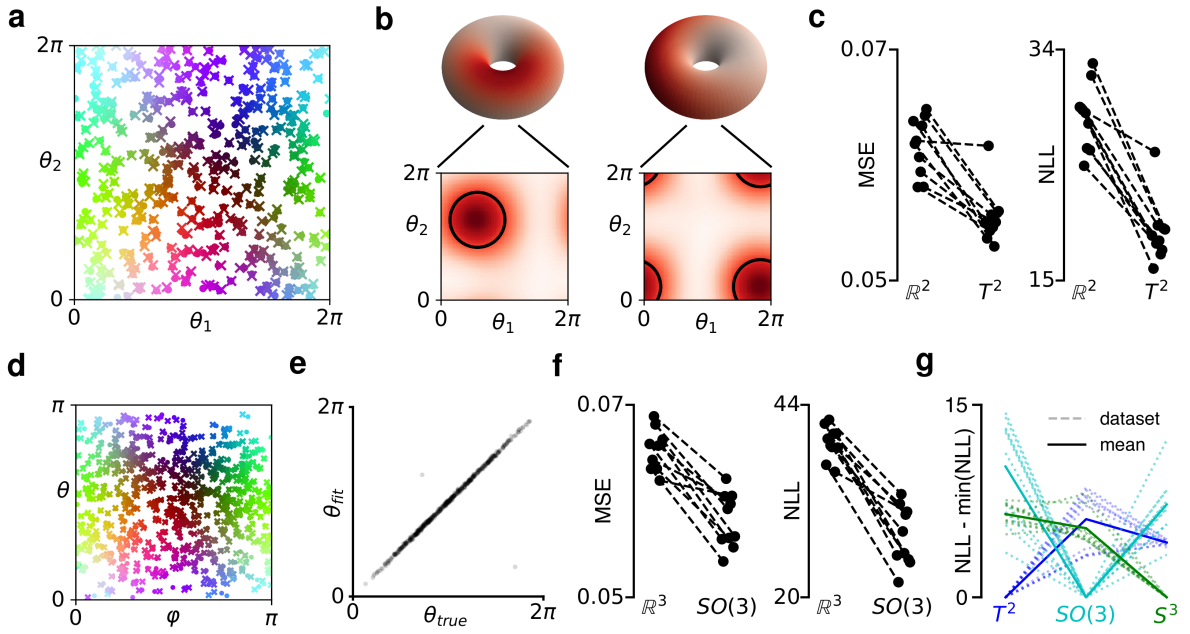


Figure 5.3: **Validating mGPLVM on synthetic data. (a-c)** Torus dataset. **(a)** True latent states $\{g_j \in T^2\}$ (dots) and posterior latent means $\{g_j^\mu\}$ (crosses). The color scheme is chosen to be smooth for the true latents. **(b)** Posterior tuning curves for two example neurons. Top: tuning curves on the tori. Bottom: projections onto the periodic $[0; 2\pi]$ plane. Black circles indicate locations and widths of the true tuning curves. **(c)** Mean squared cross-validated prediction error (left) and negative log likelihood (right) when fitting T^2 and $\mathbb{R}^2$ to data generated on T^2. Dashed lines connect datapoints for the same synthetic dataset. **(d-f)** $SO(3)$ dataset. **(d)** Axis of the rotation represented by the true latent states $\{g_j \in SO(3)\}$ (dots) and the posterior latent means $\{g_j^\mu\}$ (crosses) projected onto the (φ, θ)-plane. **(e)** Magnitude of the rotations represented by $\{g_j\}$ and $\{g_j^\mu\}$. **(f)** Same as (c), now comparing $SO(3)$ to $\mathbb{R}^3$. **(g)** Test log likelihood ratio for 10 synthetic datasets on T^2, $SO(3)$, & S^3, with mGPLVM fitted on each manifold (x-axis). Solid lines indicate mean across datasets.

fitted an $SO(3)$-mGPLVM to synthetic data generated on $SO(3)$ and found that it rendered a faithful representation of the latent space and outperformed a Euclidean GPLVM on predictions (Figure 5.3d-f). Finally we show that mGPLVM can also be used to select between multiple non-Euclidean topologies. We generated 10 datasets on each of T^2, $SO(3)$ and S^3 and compared cross-validated log likelihoods for T^2-, $SO(3)$- and S^3-mGPLVM, noting that $p(\mathcal{M}|\boldsymbol{Y}) \propto p(\boldsymbol{Y}|\mathcal{M})$ under a uniform prior over manifolds $\mathcal{M}$. Here we found that the correct latent manifold was consistently the most likely for all 30 datasets (Figure 5.3g). In summary, these results show robust performance of mGPLVM across various manifolds of interest in neuroscience and beyond, as well as a quantitative advantage over Euclidean GPLVMs which ignore the underlying topology

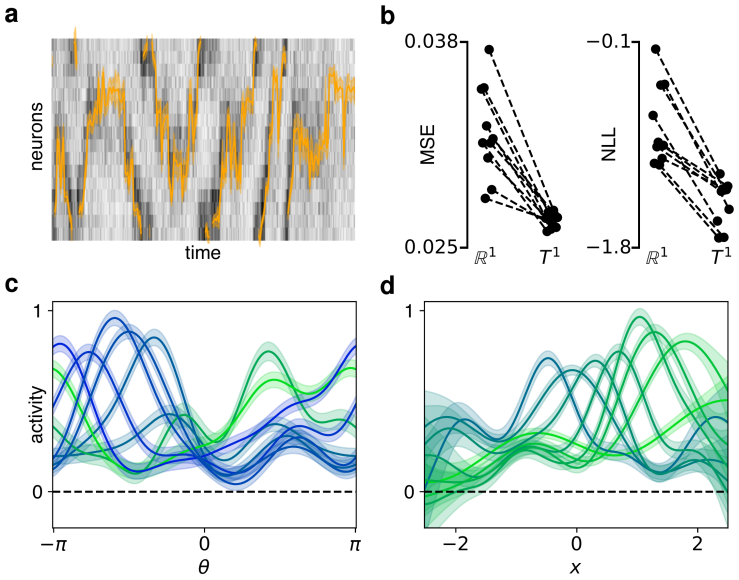


Figure 5.4: **The *Drosophila* head direction circuit. (a)** Input data overlaid with the posterior variational distribution over latent states of a T^1-mGPLVM. **(b)** Mean cross-validated prediction error (left) and negative log likelihood (right) for models fitted on T^1 and $\mathbb{R}^1$. Each datapoint corresponds to a different partition of the timepoints into a training set and a test set. **(c-d)** Posterior tuning curves for eight example neurons in T^1 (c) and $\mathbb{R}^1$ (d). Color encodes the position of the maximum of each tuning curve. Shadings in (a,c,d) indicate ±2 s.t.d.

of the latent space.

5.3.2 The *Drosophila* head direction circuit

Finally we applied mGPLVM to an experimental dataset to show that it is robust to biological and measurement noise. Here, we used calcium imaging data recorded from the ellipsoid body (EB) of *Drosophila melanogaster* [191], where the so-called E-PG neurons have recently been shown to encode head direction [162]. The EB is divided into 16 'wedges', each containing 2-3 E-PG neurons that are not distinguishable on the basis of calcium imaging data, and we therefore treat each wedge as one 'neuron'. Due to the physical shape of the EB, neurons come 'pre-ordered' since their joint activity resembles a bump rotating on a ring (Figure 5.4a, analogous to Figure 5.2, "ordered data"). While the EB's apparent ring topology obviates the need for mGPLVM as an explorative tool for uncovering manifold representations, we emphasize that head direction circuits in higher organisms are not so obviously structured (Chaudhuri et al. [17]; Section 5.5.1) – in fact, some brain areas such as the entorhinal cortex even embed concurrent representations of multiple spaces [24, 64].

We fitted the full mGPLVM with a separate GP for each neuron and found that T^1-mGPLVM

performed better than $\mathbb{R}^1$-mGPLVM on both cross-validated prediction errors and log marginal likelihoods (Figure 5.4b). The model recovered latent angles that faithfully captured the visible rotation of the activity bump around the EB, with larger uncertainty during periods where the neurons were less active (Figure 5.4a, orange). When querying the posterior tuning curves from a fit in $\mathbb{R}^1$, these were found to suffer from spurious boundary conditions with inflated uncertainty at the edges of the latent representation – regions where $\mathbb{R}^1$-mGPLVM effectively has less data than T^1-mGPLVM since $\mathbb{R}^1$ does not wrap around. In comparison, the tuning curves were more uniform across angles in T^1 which correctly captures the continuity of the underlying manifold. In Section 5.5.1, we describe similar results with mGPLVM applied to a dataset from the mouse head-direction circuit with more heterogeneous neuronal tuning and no obvious anatomical organization [140].

5.4 Discussion and future work

Conclusion We have presented an extension of the popular GPLVM model to incorporate non-Euclidean latent spaces. This is achieved by combining a Bayesian GPLVM with recently developed methods for approximate inference in non-Euclidean spaces and a new family of manifold-specific kernels. Inference is performed using variational sparse GPs for computational tractability with inducing points optimized directly on the manifold. We demonstrated that mGPLVM correctly infers the latent states and GP parameters for synthetic data of various dimensions and topologies, and that cross-validated model comparisons can recover the correct topology of the space. Finally, we showed how mGPLVM can be used to infer latent topologies and representations in biological circuits from calcium imaging data. We expect mGPLVM to be particularly valuable to the neuroscience community because many quantities encoded in the brain naturally live in non-Euclidean spaces [17, 45, 204].

Related work GP-based latent variable models with periodicity in the latent space have previously been used for motion capture, tracking and animation [36, 193]. However, these approaches are not easily generalized to other non-Euclidean topologies and do not provide a tractable marginal likelihood which forms the basis of our Bayesian model comparisons. Additionally, methods have been developed for analysing the geometry of the latent space of GPLVMs [188] and other latent variable models [6] after initially learning the models with a Euclidean latent. These approaches confer a degree of interpretability to the learned latent space but do not explicitly incorporate priors and topological constraints on the manifold during learning.

Furthermore, GPs and GPLVMs with non-Euclidean outputs have been developed [108, 109, 124]. These approaches are orthogonal to mGPLVM where the latent GP inputs, not outputs, live on a non-Euclidean manifold. mGPLVM can potentially be combined with these approaches to model non-Euclidean observations, and to incorporate more expressive GP priors over the latent states than the independent prior we have used here.

Finally, several methods for inference in non-Euclidean spaces have been developed in the machine learning literature. These have centered around methods based on VAEs [28, 146, 199], normalizing flows [148], and neural ODEs [42, 104, 114]. While non-Euclidean VAEs are useful for amortized inference, they constrain $f(g)$ more than a GP does and do not naturally allow expression of a prior over its smoothness. Normalizing flows and neural ODEs can potentially be combined with mGPLVM to increase the expressiveness of the variational distributions [41]. This would allow us to model complex distributions over latents, such as the multimodal distributions that naturally arise in ambiguous environments with symmetries [78].

mGPLVM extensions Here, we have assumed statistical independence across latent states, but prior dependencies could be introduced to incorporate e.g. temporal smoothness by placing a GP prior on the latents as in GPFA [213]. To capture more statistical structure in the latents, richer variational approximations of the posterior could be learned by using normalizing flows on the base distribution (r_θ). It would also be interesting to exploit automatic relevance determination (ARD, 125) in mGPLVM to automatically select the latent manifold dimension. We explored this approach by fitting a T^2-mGPLVM to the data from Figure 5.2 with separate lengthscales for the two dimensions, where we found that T^2 shrunk to T^1, the true underlying manifold (Section 5.5.8).

Furthermore, the mGPLVM framework can be extended to direct products of manifolds, enabling the study of brain areas encoding non-Euclidean variables such as head direction jointly with global modulation parameters such as attention or velocity. As an example, fitting a $(T^1 \times \mathbb{R}^1)$-mGPLVM to the *Drosophila* data captures both the angular heading in the T^1 dimension as well as a variable correlated with global activity in the $\mathbb{R}^1$ dimension (Section 5.5.9).

Future applications mGPLVM not only infers the most likely latent states but also estimates the associated uncertainty, which can be used as a proxy for the degree of momentary coherence expressed in neural representations. It would be interesting to compare such posterior uncertainties and tuning properties in animals across brain states. For example, uncertainty estimates could be compared across sleep and wakefulness or environments with reliable and noisy spatial cues.

In the motor domain, mGPLVM can help elucidate the neural encoding of motor plans for movements naturally specified in rotational spaces. Examples include 3-dimensional head rotations represented in the rodent superior colliculus [113, 204] as well as analogous circuits in primates. Finally, it will be interesting to apply mGPLVM to artificial agents trained on tasks that require them to form internal representations of non-Euclidean environmental variables [7]. Our framework could be used to dissect such representations, adding to a growing toolbox for the analysis of artificial neural networks [177].

5.5 Supplementary

5.5.1 The mouse head direction circuit

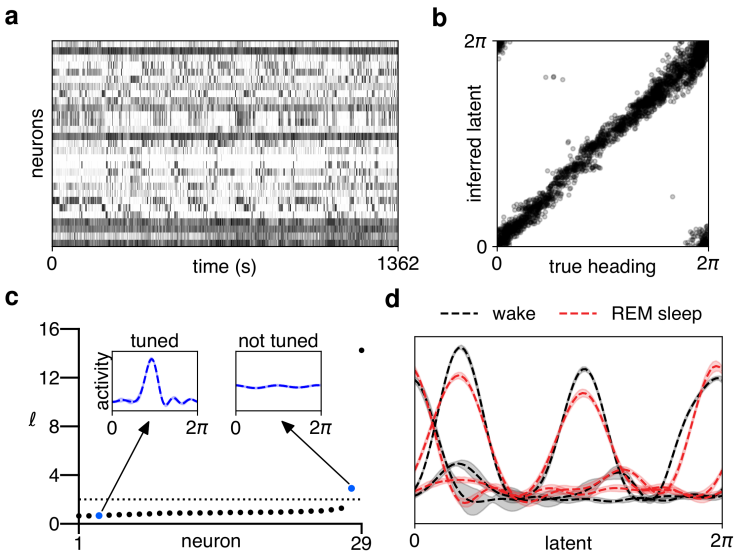


Figure 5.5: **The mouse head direction circuit. (a)** Population activity recorded from mouse ADn during foraging. **(b)** Variational mean inferred by T^1-mGPLVM plotted against the true mouse head direction. **(c)** Kernel length scales for the 29 neurons recorded. Dashed line: $\ell^2 = 4$ (maximum d in the T^1-kernel). Insets: example neurons with low and high ℓ. **(d)** Tuning curves for three example neurons inferred during wake (black) and REM sleep (red).

To highlight the importance of unsupervised non-Euclidean learning methods in neuroscience and to illustrate the interpretability of the learned GP parameters, we consider a dataset from Peyrache et al. [140] recorded from the mouse anterodorsal thalamic nucleus (ADn; Figure 5.5a). This data has also been analyzed in Peyrache et al. [139], Chaudhuri et al. [17] and Rubin et al. [152]. We consider the same example session shown in Figure 2 of Chaudhuri et al. [17] (Mouse 28, session 140313) and bin spike counts in 500 ms time bins for analysis with mGPLVM. When comparing cross-validated log likelihoods for T^1- and $\mathbb{R}^1$-mGPLVM fitted to the data, T^1 consistently outperformed $\mathbb{R}^1$ with a log likelihood ratio of 127 ± 30 (mean $\pm$ sem) across 10 partitions of the data.

Fitting T^1-mGPLVM to the binned spike data, we found that the inferred latent state was highly correlated with the true head direction (Figure 5.5b). However, in contrast to the data considered in Section 5.3.1 and Section 5.3.2, this mouse dataset contains neurons with more heterogeneous baseline activities and tuning properties. This is reflected in the learned GP parameters which converge to small kernel length scales for neurons that contribute to the heading representation (Figure 5.5c, 'tuned') and large length scales for those that do not (Figure 5.5c, 'not tuned'). Finally, since mGPLVM does not require knowledge of behaviour, we also fitted mGPLVM to data recorded from the same neurons during a period of rapid eye movement (REM) sleep. Here we found that the representation of subconscious heading during REM sleep was similar to the representation of heading when the animal was awake after matching the offset between the two sets of tuning curves (Figure 5.5d), similar to results by Peyrache et al. [139]. However, their analyses relied on recordings from two separate brain regions to align the activity from neurons in ADn to a subconscious head direction decoded from the postsubiculum and vice versa. In contrast, mGPLVM allows for fully unsupervised Bayesian analyses across both wake and sleep using recordings from a single brain area.

5.5.2 Priors on manifolds

For all manifolds, we use priors that factorize over conditions, $p^{\mathcal{M}}(\{g_j\}) = \prod_j p^{\mathcal{M}}(g_j)$. As described in Section 5.2.1, we use a Gaussian prior $p^{R^n}(g) = \mathcal{N}(g; 0, \mathbf{I}_n)$ over latent states in $\mathbb{R}^n$, and uniform priors for the spheres, tori, and $SO(3)$. These uniform priors have a density which is the inverse volume of the manifold:

$$p^{S^n}(g) = \left[\frac{2\pi^{\frac{n+1}{2}}}{\Gamma(\frac{n+1}{2})}\right]^{-1} \tag{5.23}$$

$$p^{T^n}(g) = [2\pi]^{-n} \tag{5.24}$$

$$p^{SO(3)}(g) = \left[\frac{2\pi^{\frac{4}{2}}}{2\Gamma(\frac{4}{2})}\right]^{-1}. \tag{5.25}$$

Note that the volume of S^n is the surface area of the n-sphere, and the volume of $SO(3)$ is half the volume of S^3.

5.5.3 Lie groups and their exponential maps

For simplicity of exposition, we have skimmed over the details of how the 'capitalized' Exponential map $\mathrm{Exp}_G : \mathbb{R}^n \to G$ is defined in Section 5.2.2, particularly in relation to the group's Lie algebra $\mathfrak{g}$. Here we make this connection more explicit. As described in the main text, the Lie algebra

$\mathfrak{g}$ of a group G is a vector space tangent to G at its identity element. The exponential map $\exp_G : \mathfrak{g} \to G$ maps elements from the Lie algebra to the group, and is conceptually distinct from the "capitalised" Exponential map defined in Section 5.2.2 which maps from $\mathbb{R}^n$ to G. However, because the Lie algebra is isomorphic to $\mathbb{R}^n$, we have found it convenient in both our exposition and our implementation to work directly with the pair $(\mathbb{R}^n, \mathrm{Exp}_G)$, instead of $(\mathfrak{g}, \exp_G)$. To expand on the connection between the two, note that we can define as in Sola et al. [170] the isomorphism $\mathrm{Hat} : \mathbb{R}^n \to \mathfrak{g}$, which maps every element in $\mathbb{R}^n$ to a distinct element in the Lie algebra $\mathfrak{g}$. Therefore, $\mathrm{Exp}_G : \mathbb{R}^n \to G$ is in fact the composition $\exp_G \circ \mathrm{Hat}$.

Manifold-specific parameterizations

Here we provide some further justification for the forms of $\tilde{q}_\theta(\tilde{g})$ provided in Equations (5.18) and (5.19) as well as the exponential maps which are used to derive these densities and are needed for optimization in Equation (5.11). For both T^n and $SO(3)$, we use Equation (5.8) from Falorsi et al. [41], which we repeat here for reference:

$$\tilde{q}_\theta(\tilde{g}) = \sum_{\boldsymbol{x} \in \mathbb{R}^n \,:\, \mathrm{Exp}_G(\boldsymbol{x}) = \tilde{g}} r_\theta(\boldsymbol{x}) |\boldsymbol{J}(\boldsymbol{x})|^{-1}. \tag{5.26}$$

In what follows, we will use $\boldsymbol{g}$ to indicate a vector representation of group element g to avoid conflicts of notation.

Note that the expressions in this section largely follow Falorsi et al. [41], but we re-write them in a different basis for ease of computational implementation.

T^n

The n-Torus T^n is the direct product of n circles, such that we can parameterize members of this group as $\boldsymbol{g} \in \mathbb{R}^n$ whose elements are all angles between 0 and 2π. Note that this is equivalent to the parameterization in Equation (5.16) except that here we denote an element on the circle by its angle, while in Equation (5.16) we denote it by a unit 2-vector for notational consistency with the other kernels. Because 1-dimensional rotations are commutative, the parameterization of the torus as a list of angles allows us to perform group operations by simple addition modulo 2π. We therefore slightly abuse notation and write the exponential map $\mathrm{Exp}_{T^n} : \mathbb{R}^n \to T^n$ as an element-wise modulo operation:

$$\mathrm{Exp}_{T^n} \boldsymbol{x} = \boldsymbol{x} \bmod 2\pi. \tag{5.27}$$

Equation (5.27) has inverse Jacobian $|\boldsymbol{J}(x)|^{-1} = 1$. Moreover, since $\mathrm{Exp}_{T^n}(\boldsymbol{x}) = \mathrm{Exp}_{T^n}(\boldsymbol{x} + 2\pi\boldsymbol{k})$ for any integer vector $\boldsymbol{k} \in \mathbb{Z}^n$, the change-of-variable formula in Equation (5.26) yields the

following density on T^n:

$$\tilde{q}_\theta(\text{Exp}_{T^n}\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^n} r_\theta(\boldsymbol{x} + 2\pi\boldsymbol{k}). \tag{5.28}$$

For ease of implementation it is also convenient to rewrite the kernel distance function Equation (5.16) as

$$d_{T^n}(\boldsymbol{g}, \boldsymbol{g}') = 2 \cdot \mathbf{1}_n \cdot (1 - \cos(\boldsymbol{g} - \boldsymbol{g}')) \tag{5.29}$$

where $\mathbf{1}_n$ is the n-vector full of ones, and $\cos(\cdot)$ is applied element-wise to $\boldsymbol{g} - \boldsymbol{g}'$.

5.5.4 $SO(3)$

We use quaternions $\boldsymbol{g} \in \mathbb{R}^4$ to represent elements $g \in SO(3)$ as indicated in Equation (5.17). For a rotation of ϕ radians around axis $\boldsymbol{u} \in \mathbb{R}^3$ with $\|\boldsymbol{u}\| = 1$,

$$\boldsymbol{g} = \left(\cos\frac{\phi}{2}, \boldsymbol{u}\sin\frac{\phi}{2}\right) \in \mathbb{R}^4. \tag{5.30}$$

The exponential map $\text{Exp}_{SO(3)} : \mathbb{R}^3 \to SO(3)$ is

$$\text{Exp}_{SO(3)}\boldsymbol{x} = (\cos\|\boldsymbol{x}\|, \hat{\boldsymbol{x}}\sin\|\boldsymbol{x}\|), \tag{5.31}$$

where $\hat{\boldsymbol{x}} = \boldsymbol{x}/\|\boldsymbol{x}\|$ and $\phi = 2\|\boldsymbol{x}\|$ is the angle of rotation. This gives rise to an inverse Jacobian

$$|\boldsymbol{J}(\boldsymbol{x})|^{-1} = \phi^2/(2(1 - \cos\phi)). \tag{5.32}$$

Using Equation (5.26) we get the density on the group

$$\tilde{q}_\theta(\text{Exp}_{SO(3)}\boldsymbol{x}) = \sum_{k\in\mathbb{Z}} \left[r_\theta(\boldsymbol{x} + \pi k\hat{\boldsymbol{x}}) \frac{2\|\boldsymbol{x} + \pi k\hat{\boldsymbol{x}}\|^2}{1 - \cos(2\|\boldsymbol{x} + \pi k\hat{\boldsymbol{x}}\|)} \right], \tag{5.33}$$

where the sum over k stems from the fact that a rotation of $\phi + 2k\pi$ around axis $\hat{\boldsymbol{x}}$ is equivalent to a rotation of ϕ around the same axis.

5.5.5 S^n

In this section, we discuss how to fit mGPLVMs on spheres. We first consider spheres which are also Lie groups, and then discuss a general framework for all n-spheres.

$S^{1,3}$

We begin by noting that S^n is not a Lie group unless $n = 1$ or $n = 3$, thus we can only apply the ReLie framework to S^1 and S^3. S^1 is equivalent to T^1 and is most easily treated using the torus formalism above. For S^3, we note that $SO(3)$ is simply S^3 with double coverage. This

is because quaternions $\boldsymbol{g}$ and $-\boldsymbol{g}$ represent the same element of $SO(3)$ while they correspond to distinct elements of S^3. The Jacobian and exponential maps of S^3 are therefore identical to those of $SO(3)$. The expression for the density on S^3 also mirrors Equation (5.33) except that the sum is over $\boldsymbol{x} + 2\pi k\hat{\boldsymbol{x}}$ instead of $\boldsymbol{x} + \pi k\hat{\boldsymbol{x}}$:

$$\tilde{q}_\theta(\mathrm{Exp}_{S^3}\boldsymbol{x}) = \sum_{k\in\mathbb{Z}} \left[r_\theta(\boldsymbol{x} + 2\pi k\hat{\boldsymbol{x}}) \, \frac{2\|\boldsymbol{x} + 2\pi k\hat{\boldsymbol{x}}\|^2}{1 - \cos\left(2\|\boldsymbol{x} + 2\pi k\hat{\boldsymbol{x}}\|\right)} \right]. \tag{5.34}$$

We demonstrate S^3-mGPLVM on synthetic data from S^3 in Figure 5.6 (bottom).

$S^{n\notin\{1,3\}}$

The ReLie framework does not directly apply to distributions defined on non-Lie groups. Nevertheless, we can still apply mGPLVM to an n-sphere embedded in $\mathbb{R}^{n+1}$ by taking each latent variational distribution q_{θ_j} to be a von Mises-Fisher distribution (VMF), whose entropy is known analytically. Parameterizing group element $g \in S^n$ by a unit-norm vector $\boldsymbol{g} \in \mathbb{R}^{n+1}$, $\|\boldsymbol{g}\| = 1$, this density is given by:

$$q_\theta(\boldsymbol{g}; \boldsymbol{g}^\mu, \kappa) = \frac{\kappa^{n/2-1}}{(2\pi)^{n/2} I_{n/2-1}(\kappa)} \exp(\kappa\, \boldsymbol{g}^\mu \cdot \boldsymbol{g}) \tag{5.35}$$

where $\cdot$ denotes the dot product. Here, I_v is the modified Bessel function of the first kind at order v, $\boldsymbol{g}^\mu$ is the mean direction of the distribution on the hypersphere, and $\kappa \geq 0$ is a concentration parameter – the larger κ, the more concentrated the distribution around $\boldsymbol{g}^\mu$.

Using a VMF distribution as the latent distribution, we can easily evaluate the ELBO in Equation (5.5) because (i) there are well-known algorithms for sampling from the distribution using rejection-sampling [192] and (ii) both the entropy term $H(q_\theta)$ and its gradient can be derived analytically [28]. For details of how to differentiate through rejection sampling, please refer to Naesseth et al. [122] and Davidson et al. [28].

In the following, we provide details for applying mGPLVM to S^2 for which we do not need to use rejection sampling and instead use inverse transform sampling [79]. For S^2, the VMF distribution simplifies to [171]

$$q_\theta(\boldsymbol{g}; \boldsymbol{g}^\mu, \kappa) = \frac{\kappa}{2\pi(\exp(\kappa) - \exp(-\kappa))} \exp(\kappa\, \boldsymbol{g}^\mu \cdot \boldsymbol{g}), \tag{5.36}$$

and its entropy is

$$H(q_\theta) = -\int_{S^2} q_\theta(\boldsymbol{g}; \boldsymbol{g}^\mu, \kappa) \log q_\theta(\boldsymbol{g}; \boldsymbol{g}^\mu, \kappa) d\boldsymbol{g} \tag{5.37}$$

$$= -\log\left(\frac{\kappa}{4\pi \sinh \kappa}\right) - \frac{\kappa}{\tanh \kappa} + 1. \tag{5.38}$$

These equations allow us to apply mGPLVM to S^2 by optimizing the ELBO as described in the main text; this is illustrated for synthetic data on S^2 in Figure 5.6 (top).

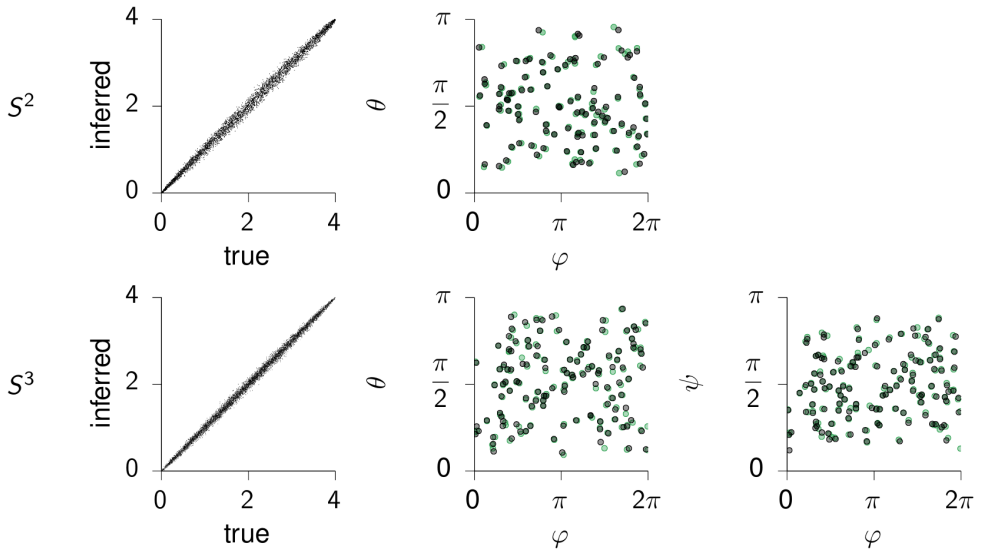


Figure 5.6: **Applying mGPLVM to synthetic data on S^2 (top) and S^3 (bottom).** Pairwise distances between the variational means $\{g_j^\mu\}$ are plotted against the corresponding pairwise distances between the true latent states $\{g_j\}$ for S^2 (top left) and S^3 (bottom left). Since the log likelihood is a function of these pairwise distances through the kernel (Equation (5.15)), this illustrates that mGPLVM recovers the important features of the true latents. Inferred (black) and true (green) latent states in spherical coordinates for S^2 (top middle) and S^3 (bottom middle and bottom right). For S^2, we are showing the latent states in spherical polar coordinates $\boldsymbol{g} = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta)$ with $\theta \in [0, \pi]$ and $\varphi \in [0, 2\pi]$. For S^3, we use hyperspherical coordinates $\boldsymbol{g} = (\sin\psi\sin\theta\cos\varphi, \sin\psi\sin\theta\sin\varphi, \sin\theta\cos\psi, \cos\theta)$ with $\theta, \psi \in [0, \pi]$ and $\varphi \in [0, 2\pi]$.

5.5.6 Posterior over tuning curves

We can derive the posterior over tuning curves in Equation (5.12) as follows:

$$p(\boldsymbol{f}_i^\star|\boldsymbol{Y}, \mathcal{G}^\star) = \int p(\boldsymbol{f}_i^\star, \mathcal{G}|\mathcal{G}^\star, \boldsymbol{Y})\, d\mathcal{G} \tag{5.39}$$

$$= \int p(\boldsymbol{f}_i^\star|\mathcal{G}^\star, \{\mathcal{G}, \boldsymbol{Y}\})p(\mathcal{G}|\boldsymbol{Y})\, d\mathcal{G} \tag{5.40}$$

$$\approx \int p(\boldsymbol{f}_i^\star|\mathcal{G}^\star, \{\mathcal{G}, \boldsymbol{Y}\})Q_\theta(\mathcal{G})\, d\mathcal{G} \tag{5.41}$$

$$\approx \frac{1}{K}\sum_{k=1}^{K} p(\boldsymbol{f}_i^\star|\mathcal{G}^\star, \{\mathcal{G}_k, \boldsymbol{Y}\}) \tag{5.42}$$

where each $\mathcal{G}_k$ is a set of M latents (one for each of the M conditions in the data $\boldsymbol{Y}$) sampled from the variational posterior $Q_\theta(\mathcal{G})$. The standard deviation around the mean tuning curves in all figures are estimated from 1000 independent samples from this posterior, with each draw involving the following two steps: (i) draw a sample $\mathcal{G}_k$ from Q_θ and (ii) conditioned on this sample, draw

from the predictive distribution $p(\boldsymbol{f}_i^\star|\mathcal{G}^\star, \{\mathcal{G}_k, \boldsymbol{Y}\})$. Together, these two steps correspond to a single draw from the posterior. Note that we make a variational sparse GP approximation (Section 5.2.2) and therefore approximate the predictive distribution $p(\boldsymbol{f}_i^\star|\mathcal{G}^\star, \{\mathcal{G}_k, \boldsymbol{Y}\})$ as described in Titsias [181].

5.5.7 Alignment for visualization

The mGPLVM solutions for non-Euclidean spaces are degenerate because the ELBO depends on the sampled latents through (i) their uniform prior density, (ii) their entropy, and (iii) the GP marginal likelihood, and all three quantities are invariant to transformations that preserve pairwise distances. For example, the application of a common group element g to *all* the variational means leaves pairwise distances unaffected and therefore does not affect the ELBO. Additionally, pairwise distances are invariant to reflections along any axis of the coordinate system we have chosen to represent each group. Therefore, to plot comparisons between true and fitted latents, we use numerical optimization to find a single distance-preserving transformation that minimizes the average geodesic distance between the variational means $\{g_j^\mu\}$ and the true latents $\{g_j\}$.

For the n-dimensional torus (Figures 5.2 and 5.3) which we parameterize as

$$\boldsymbol{g} \in \{(g_1, \cdots, g_n); \forall k : g_k \in [0, 2\pi]\},$$

the distance metric depends on $\cos(g_k - g'_k)$ and is invariant to any translation and reflection of all latents along each dimension

$$g_k \to (\alpha_k g_k + \beta_k) \mod 2\pi$$

where $\alpha_k \in \{1, -1\}$ and $\beta_k \in [0, 2\pi]$. We optimize discretely over the $\{\alpha_k\}$ by trying every possible combination, and continuously over β_k for each combination of $\{\alpha_k\}$.

In the case of S^2, S^3 and $SO(3)$ (Figures 5.3 and 5.6), the distance metrics are invariant to unitary transformations $\boldsymbol{g} \to \boldsymbol{R}\boldsymbol{g}$ where $\boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}$ for the parameterizations used in this work. For visualization of these groups, we align the inferred latents with the true latents by optimizing over $\boldsymbol{R}$ on the manifold of orthogonal matrices.

5.5.8 Automatic relevance determination

As we mention in Section 5.4, it is possible to exploit automatic relevance determination (ARD) for automatic selection of the dimensionality of groups with additive distance metrics such as

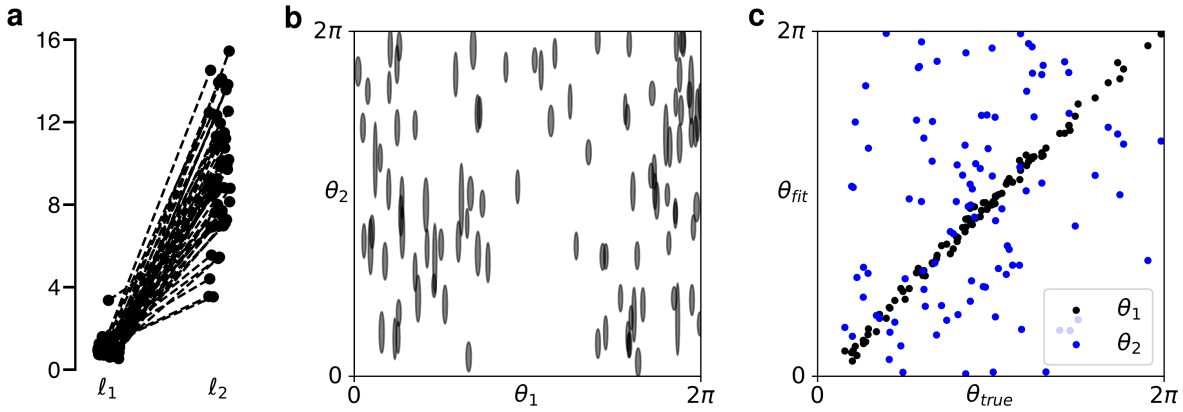


Figure 5.7: **Automatic relevance determination (ARD) in T^2-mGPLVM.** A T^2 model with ARD was fitted to the T^1 data in Figure 5.2. **(a)** Length scales along each of the two dimensions for each neuron. **(b)** Posterior variational distributions. Shading indicates ± 1 s.t.d. around the posterior mean in each dimension. **(c)** Variational mean plotted against the true latent state for each dimension.

the T^n-distance in Equation (5.29). While we have not investigated this in detail, we illustrate the idea here on a simple example. We consider the same synthetic data as in Figure 5.2 and fit a T^2-mGPLVM with a kernel on T^2 that has separate lengthscales ℓ_1 and ℓ_2 for each dimension:

$$k_{T^2_{\text{ARD}}}(\boldsymbol{g}, \boldsymbol{g}') = \alpha^2 \exp\left(\frac{\cos(g_1 - g_1') - 1}{\ell_1^2}\right) \exp\left(\frac{\cos(g_2 - g_2') - 1}{\ell_2^2}\right). \tag{5.43}$$

Additionally, we assume the variational distribution to factorize across latent dimensions:

$$q_{\theta_j}(\cdot) = q_{\theta_j^1}(\cdot)\, q_{\theta_j^2}(\cdot), \tag{5.44}$$

such that their entropies add up to the total entropy:

$$H(q_{\theta_j}) = H(q_{\theta_j^1}) + H(q_{\theta_j^2}). \tag{5.45}$$

This corresponds to assuming that each variational covariance matrix $\boldsymbol{\Sigma}_j$ (Section 5.2.2) is diagonal.

When fitting this model, we find that one length parameter goes to large values while the other remains on the order of the size of the space (Figure 5.7a; note that $d_{T^1} \in [0, 4]$). This indicates that neurons are only tuned to one of the two torus dimensions. Additionally, posterior variances become very large in the non-contributing dimension, i.e. the data does not contain the other angular dimension (Figure 5.7b). This further indicates that the model has effectively shrunk from a 2-torus to a single circle. We note that the entropy of the factor in the variational posterior that corresponds to the discarded dimension becomes $\log 2\pi$ as the variance goes to infinity in this direction. This exactly offsets the increased complexity penalty of the prior for T^2 compared

to T^1, such that the two models have the same ELBO. The model thus reduces to a T^1 model, demonstrating how ARD can be exploited to automatically infer the dimensionality of the latent space.

5.5.9 Direct products of Lie groups

Here, we elaborate slightly on the extension of mGPLVM to direct products of Lie groups, briefly mentioned in the discussion (Section 5.4). Assuming additive distance metrics and factorizable variational distributions, direct product kernels become multiplicative and entropies become additive – very much as in our illustration of ARD in Section 5.5.8. That is, for a group product $\mathcal{M} = \mathcal{M}_1 \times \ldots \times \mathcal{M}_L$, we can write

$$k^{\mathcal{M}}(g, g') = \prod_l k^{\mathcal{M}_l}(g, g'), \tag{5.46}$$

$$H(q_{\theta_j}^{\mathcal{M}}) = \sum_l H(q_{\theta_j}^{\mathcal{M}_l}). \tag{5.47}$$

As a simple example, we consider a $(T^1 \times \mathbb{R}^1)$-mGPLVM which we fit to the *Drosophila* data from Section 5.3.2. Here we find that the T^1 dimension of the group product, which we denote by $\theta^{(T^1 \times \mathbb{R}^1)}$, captures the angular component of the data since it is very strongly correlated with the latent state θ^{T^1} inferred by the simpler T^1-mGPLVM (Figure 5.8a). It is somewhat harder to predict what features of the data will be captured by the $\mathbb{R}^1$ dimension $x^{(T^1 \times \mathbb{R}^1)}$ of the $(T^1 \times \mathbb{R}^1)$-mGPLVM, but we hypothesize that it might capture a global temporal modulation of the neural activity. We therefore plot the mean instantaneous activity $\overline{y}$ across neurons against $x^{(T^1 \times \mathbb{R}^1)}$ and find that these quantities are indeed positively correlated (Figure 5.8b). This exemplifies how an mGPLVM on a direct product of groups can capture qualitatively different components of the data by combining representations with different topologies.

This direct product model is very closely related to the ARD model in Section 5.5.8, and the two can also be combined in a direct product of ARD kernels. For example, we can imagine constructing a $(T^n \times \mathbb{R}^n)$ direct product ARD kernel which automatically selects the appropriate number of both periodic and scalar dimensions that best, and most parsimoniously, explains the data.

5.5.10 Implementation

Scaling As mentioned in Section 5.2.2, approximating the GP likelihood term $\mathbb{E}_{Q_\theta}[\log p(\boldsymbol{Y}|\{g_j\})]$ in the mGPLVM ELBO scales as $\mathcal{O}(m^2MNK)$ with m inducing points, M latent states, N neurons, and K Monte Carlo samples. Estimating the entropy term is $\mathcal{O}(MKd)$ for a d-dimensional

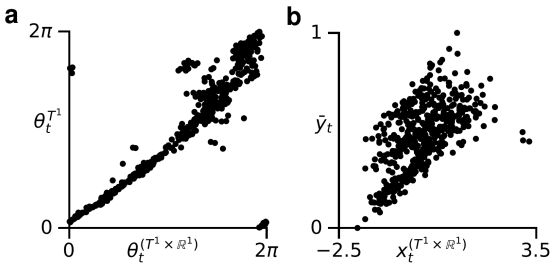


Figure 5.8: $(T^1 \times \mathbb{R}^1)$**-mGPLVM. (a)** Latent states inferred by T^1-mGPLVM (Figure 5.4a) against the periodic coordinate of a $(T^1 \times \mathbb{R}^1)$-mGPLVM fitted to the *Drosophila* data. **(b)** Momentary average population activity $\bar{y}_t$ against the scalar Euclidean component of the $(T^1 \times \mathbb{R}^1)$ latent representation.

Euclidean latent space, $\mathcal{O}(MK(2k_{max}+1)^d)$ for a d-dimensional torus, and $\mathcal{O}(MK(2k_{max}+1))$ for $SO(3)$ and S^3, where k_{max} is the maximum value of k used in Equation (5.8). For all manifolds considered in this work, we can compute a closed-form Exp$(\cdot)$ while for general matrix Lie groups, approximating Exp as a power series is $\mathcal{O}(d^3)$ [41], further increasing the complexity of mGPLVM for such groups.

For our manifolds of interest, computing the likelihood term tends to be the main computational bottleneck, although the entropy term can become prohibitive for high-dimensional periodic latents [148]. When computing $\mathbb{E}_{Q_\theta}[\log p(\boldsymbol{Y}|\{g_j\})]$, most of the complexity is due to inverting NK matrices of size $(Mm^2) \times (Mm^2)$, which can be performed in parallel for each Monte Carlo sample and neuron. Using PyTorch for parallelization across neurons and MC samples, we can train T^1-mGPLVM with $N = 300$ and $M = 1000$ in $\sim$100 seconds on an NVIDIA GeForce RTX 2080 GPU with 8GB RAM.

Initialization For all simulations, we initialized the system with variational means at the identity element of the manifold, but with large variational variances to reflect the lack of prior information about the true latent states. Inducing points were initialized according to the prior on each manifold (Equation (5.1)). To avoid variational distributions collapsing to the uniform distribution early during learning, we ran a preliminary 'warm up' optimization phase during which some of the parameters were held fixed. Specifically, we fixed the variational covariance matrices as well as the kernel variance parameters (α in Equation (5.13)), and prioritized a better data fit by setting the entropy term to zero in Equation (5.5). Learning proceeded as normal thereafter.

Entropy approximation When evaluating Equation (5.8), we used values of $k_{max} = 3$ for the tori and S^3 as in Falorsi et al. [41] and $k_{max} = 5$ for $SO(3)$ since the sum takes steps of π instead of 2π. In theory, the finite k_{max} can lead to an overestimation of the ELBO for large

variational uncertainties, as $\tilde{q}$ is systematically underestimated, leading to overestimation of the entropy. To mitigate this, we capped the approximate entropy for non-Euclidean manifolds at the maximum entropy corresponding to a uniform distribution on the manifold.

Bibliography

[1] Afshar, A., Santhanam, G., Byron, M. Y., Ryu, S. I., Sahani, M., and Shenoy, K. V. (2011). Single-trial neural correlates of arm movement preparation. *Neuron*, 71:555–564.

[2] Aitchison, L. and Lengyel, M. (2016). The hamiltonian brain: efficient probabilistic inference with excitatory-inhibitory neural circuit dynamics. *PLoS Comput. Biol.*, 12:e1005186.

[3] Ames, K. C., Ryu, S. I., and Shenoy, K. V. (2014). Neural dynamics of reaching following incorrect or absent motor preparation. *Neuron*, 81:438–451.

[4] Anderson, B. D. and Moore, J. B. (2007). *Optimal control: linear quadratic methods*. Courier Corporation.

[5] Andersson, J. A., Gillis, J., Horn, G., Rawlings, J. B., and Diehl, M. (2019). CasADi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36.

[6] Arvanitidis, G., Hansen, L. K., and Hauberg, S. (2017). Latent space oddity: on the curvature of deep generative models. *arXiv preprint arXiv:1710.11379*.

[7] Banino, A., Barry, C., Uria, B., Blundell, C., Lillicrap, T., Mirowski, P., Pritzel, A., Chadwick, M. J., Degris, T., Modayil, J., Wayne, G., Soyer, H., Viola, F., Zhang, B., Goroshin, R., Rabinowitz, N., Pascanu, R., Beattie, C., Petersen, S., Sadik, A., Gaffney, S., King, H., Kavukcuoglu, K., Hassabis, D., Hadsell, R., and Kumaran, D. (2018). Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433.

[8] Barak, O. (2017). Recurrent neural networks as versatile tools of neuroscience research. *Curr. Op. Neurobiol.*, 46:1–6.

[9] Bartels, R. H. and Stewart, G. W. (1972). Solution of the matrix equation AX+XB=C. *Commun. ACM*, 15:820–826.

[10] Baydin, A. G., Pearlmutter, B. A., Radul, A. A., and Siskind, J. M. (2017). Automatic differentiation in machine learning: a survey. *J. Mach. Learn. Res.*, 18(1):5595–5637.

[11] Bettencourt, J., Johnson, M. J., and Duvenaud, D. (2019). Taylor-mode automatic differentiation for higher-order derivatives in jax.

[12] Borovitskiy, V., Terenin, A., Mostowsky, P., and Deisenroth, M. P. (2020). Matérn Gaussian processes on Riemannian manifolds. *arXiv preprint arXiv:2006.10160*.

[13] Buesing, L., Macke, J. H., and Sahani, M. (2012). Learning stable, regularised latent models of neural population dynamics. *Netw. Comput. Neural Syst.*, 23:24–47.

[14] Burda, Y., Grosse, R., and Salakhutdinov, R. (2015). Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*.

[15] Byrd, R. H., Lu, P., Nocedal, J., and Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16:1190–1208.

[16] Carnevale, F., de Lafuente, V., Romo, R., Barak, O., and Parga, N. (2015). Dynamic control of response criterion in premotor cortex during perceptual detection under temporal uncertainty. *Neuron*, 86:1067–1077.

[17] Chaudhuri, R., Gerçek, B., Pandey, B., Peyrache, A., and Fiete, I. (2019). The intrinsic attractor manifold and population dynamics of a canonical cognitive circuit across waking and sleep. *Nat. Neurosci.*, 22(9):1512–1520.

[18] Chettih, S. N. and Harvey, C. D. (2019). Single-neuron perturbations reveal feature-specific competition in v1. *Nature*, 567:334–340.

[19] Churchland, M. M., Afshar, A., and Shenoy, K. V. (2006). A central source of movement variability. *Neuron*, 52:1085–1096.

[20] Churchland, M. M., Byron, M. Y., Cunningham, J. P., Sugrue, L. P., Cohen, M. R., Corrado, G. S., Newsome, W. T., Clark, A. M., Hosseini, P., Scott, B. B., et al. (2010a). Stimulus onset quenches neural variability: a widespread cortical phenomenon. *Nat. Neurosci.*, 13:369–378.

[21] Churchland, M. M., Cunningham, J. P., Kaufman, M. T., Foster, J. D., Nuyujukian, P., Ryu, S. I., and Shenoy, K. V. (2012). Neural population dynamics during reaching. *Nature*, 487(7405):51–56.

[22] Churchland, M. M., Cunningham, J. P., Kaufman, M. T., Ryu, S. I., and Shenoy, K. V. (2010b). Cortical preparatory activity: representation of movement or first cog in a dynamical machine? *Neuron*, 68:387–400.

[23] Churchland, M. M. and Shenoy, K. V. (2007). Temporal complexity and heterogeneity of single-neuron activity in premotor and motor cortex. *J. Neurophysiol.*, 97:4235–4257.

[24] Constantinescu, A. O., O'Reilly, J. X., and Behrens, T. E. (2016). Organizing conceptual knowledge in humans with a gridlike code. *Science*, 352:1464–1468.

[25] Cui, G., Jun, S. B., Jin, X., Pham, M. D., Vogel, S. S., Lovinger, D. M., and Costa, R. M. (2013). Concurrent activation of striatal direct and indirect pathways during action initiation. *Nature*, 494:238–242.

[26] Cunningham, J. P. and Ghahramani, Z. (2015). Linear dimensionality reduction: Survey, insights, and generalizations. *J. Mach. Learn. Res.*, 16:2859–2900.

[27] Cunningham, J. P. and Yu, B. M. (2014). Dimensionality reduction for large-scale neural recordings. *Nat. Neurosci.*, 17(11):1500–1509.

[28] Davidson, T. R., Falorsi, L., De Cao, N., Kipf, T., and Tomczak, J. M. (2018). Hyperspherical variational auto-encoders. *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence*.

[29] Dayan, P. and Abbott, L. F. (2001). *Theoretical neuroscience*. Cambridge, MA: MIT Press.

[30] Delp, S. L., Anderson, F. C., Arnold, A. S., Loan, P., Habib, A., John, C. T., Guendelman, E., and Thelen, D. G. (2007). Opensim: open-source software to create and analyze dynamic simulations of movement. *IEEE Trans. Biomed. Eng.*, 54(11):1940–1950.

[31] Desmurget, M. and Grafton, S. (2000). Forward modeling allows feedback control for fast reaching movements. *Trends Cogn. Sci.*, 4(11):423–431.

[32] Dudman, J. T. and Krakauer, J. W. (2016). The basal ganglia: from motor commands to the control of vigor. *Curr. Opin. Neurobiol.*, 37:158–166.

[33] Duncker, L., O'Shea, D., Goo, W., Shenoy, K., and Sahani, M. (2017). Low-rank non-stationary population dynamics can account for robustness to optogenetic stimulation. In *Cosyne, Salt Lake City, UT. T-29.*

[34] Economo, M. N., Viswanathan, S., Tasic, B., Bas, E., Winnubst, J., Menon, V., Graybuck, L. T., Nguyen, T. N., Smith, K. A., Yao, Z., et al. (2018). Distinct descending motor cortex pathways and their roles in movement. *Nature*, 563:79–84.

[35] Egnor, S. R. and Branson, K. (2016). Computational analysis of behavior. *Annu. Rev. Neurosci.*, 39:217–236.

[36] Elgammal, A. and Lee, C.-S. (2008). Tracking people on a torus. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(3):520–538.

[37] Elsayed, G. F., Lara, A. H., Kaufman, M. T., Churchland, M. M., and Cunningham, J. P. (2016). Reorganization between preparatory and movement population responses in motor cortex. *Nat. Commun.*, 7:1–15.

[38] Emiliani, V., Cohen, A. E., Deisseroth, K., and Häusser, M. (2015). All-optical interrogation of neural circuits. *J. Neurosci.*, 35:13917–13926.

[39] Evarts, E. V. (1964). Temporal patterns of discharge of pyramidal tract neurons during sleep and waking in the monkey. *J. Neurophysiol.*, 27(2):152–171.

[40] Evarts, E. V. (1968). Relation of pyramidal tract activity to force exerted during voluntary movement. *J. Neurophysiol.*, 31(1):14–27.

[41] Falorsi, L., de Haan, P., Davidson, T. R., and Forré, P. (2019). Reparameterizing distributions on Lie groups. *arXiv preprint arXiv:1903.02958*.

[42] Falorsi, L. and Forré, P. (2020). Neural ordinary differential equations on manifolds. *arXiv preprint arXiv:2006.06663*.

[43] Feragen, A., Lauze, F., and Hauberg, S. (2015). Geodesic exponential kernels: When curvature and linearity conflict. In *Proc. IEEE Comput. Soc. Conf. Vis. Pattern Recognit.*, pages 3032–3042.

[44] Ferguson, K. A. and Cardin, J. A. (2020). Mechanisms underlying gain modulation in the cortex. *Nat. Rev. Neurosci.*, 21:80–92.

[45] Finkelstein, A., Derdikman, D., Rubin, A., Foerster, J. N., Las, L., and Ulanovsky, N. (2015). Three-dimensional head-direction coding in the bat brain. *Nature*, 517(7533):159–164.

[46] Friedrich, R. W. and Laurent, G. (2001). Dynamic optimization of odor representations by slow temporal patterning of mitral cell activity. *Science*, 291:889–894.

[47] Gabbiani, F. and Cox, S. J. (2017). *Mathematics for neuroscientists*. Academic Press.

[48] Gallego, J. A., Perich, M. G., Miller, L. E., and Solla, S. A. (2017). Neural manifolds for the control of movement. *Neuron*, 94:978–984.

[49] Ganguli, S., Bisley, J. W., Roitman, J. D., Shadlen, M. N., Goldberg, M. E., and Miller, K. D. (2008a). One-dimensional dynamics of attention and decision making in lip. *Neuron*, 58:15–25.

[50] Ganguli, S., Huh, D., and Sompolinsky, H. (2008b). Memory traces in dynamical systems. *Proc. Natl. Acad. Sci. U.S.A.*, 105:18970–18975.

[51] Gao, P., Trautmann, E., Yu, B., Santhanam, G., Ryu, S., Shenoy, K., and Ganguli, S. (2017). A theory of multineuronal dimensionality, dynamics and measurement. *BioRxiv*, pages 1–20.

[52] Gao, Y., Buesing, L., Shenoy, K. V., and Cunningham, J. P. (2015). High-dimensional neural spike train analysis with generalized count linear dynamical systems. In *Adv. Neural Inf. Process. Syst.*, pages 2044–2052.

[53] Gao, Z., Davis, C., Thomas, A. M., Economo, M. N., Abrego, A. M., Svoboda, K., De Zeeuw, C. I., and Li, N. (2018). A cortico-cerebellar loop for motor planning. *Nature*, 563:113–116.

[54] Georgopoulos, A. P. (1991). Higher order motor control. *Annu. Rev. Neurosci.*, 14(1):361–377.

[55] Georgopoulos, A. P., Kalaska, J. F., Caminiti, R., and Massey, J. T. (1982). On the relations between the direction of two-dimensional arm movements and cell discharge in primate motor cortex. *J. Neurosci.*, 2(11):1527–1537.

[56] Giles, M. B. (2008). Collected matrix derivative results for forward and reverse mode algorithmic differentiation. In *Advances in Automatic Differentiation*, pages 35–44. Springer.

[57] Gillis, J. (2015). *Practical methods for approximate robust periodic optimal control of nonlinear mechanical systems*. PhD thesis, KU Leuven.

[58] Goldman, M. S. (2009). Memory without feedback in a neural network. *Neuron*, 61:621–634.

[59] Golub, M. D., Sadtler, P. T., Oby, E. R., Quick, K. M., Ryu, S. I., Tyler-Kabara, E. C., Batista, A. P., Chase, S. M., and Yu, B. M. (2018). Learning by neural reassociation. *Nat. Neurosci.*, 21:607–616.

[60] Golub, M. D., Yu, B. M., and Chase, S. M. (2015). Internal models for interpreting neural population activity during sensorimotor control. *eLife*, 4:e10015.

[61] Griffin, D. M., Hoffman, D. S., and Strick, P. L. (2015). Corticomotoneuronal cells are "functionally tuned". *Science*, 350(6261):667–670.

[62] Gross, C. G. (2007). The discovery of motor cortex and its background. *J. Hist. Neurosci.*, 16(3):320–331.

[63] Guo, Z. V., Inagaki, H. K., Daie, K., Druckmann, S., Gerfen, C. R., and Svoboda, K. (2017). Maintenance of persistent activity in a frontal thalamocortical loop. *Nature*, 545(7653):181–186.

[64] Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., and Moser, E. I. (2005). Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436:801–806.

[65] Hahn, J. and Edgar, T. F. (2002). An improved method for nonlinear model reduction using balancing of empirical gramians. *Comput. Chem. Eng.*, 26:1379–1397.

[66] Halassa, M. M. and Acsády, L. (2016). Thalamic inhibition: diverse sources, diverse scales. *Trends. Neurosci.*, 39:680–693.

[67] Halassa, M. M. and Sherman, S. M. (2019). Thalamocortical circuit motifs: a general framework. *Neuron*, 103:762–770.

[68] Hardcastle, K., Maheswaranathan, N., Ganguli, S., and Giocomo, L. M. (2017). A multiplexed, heterogeneous, and adaptive code for navigation in medial entorhinal cortex. *Neuron*, 94:375–387.

[69] Haykin, S. (2004). *Kalman filtering and neural networks*, volume 47. John Wiley & Sons.

[70] Hennequin, G., Agnes, E. J., and Vogels, T. P. (2017). Inhibitory plasticity: balance, control, and codependence. *Ann. Rev. Neurosci.*, 40:557–579.

[71] Hennequin, G., Ahmadian, Y., Rubin, D. B., Lengyel, M., and Miller, K. D. (2018). The dynamical regime of sensory cortex: stable dynamics around a single stimulus-tuned attractor account for patterns of noise variability. *Neuron*, 98:846–860.

[72] Hennequin, G., Aitchison, L., and Lengyel, M. (2014a). Fast sampling-based inference in balanced neuronal networks. In *Adv. Neural Inf. Process. Syst.*, pages 2240–2248.

[73] Hennequin, G., Vogels, T. P., and Gerstner, W. (2014b). Optimal control of transient dynamics in balanced networks supports generation of complex movements. *Neuron*, 82:1394–1406.

[74] Hennig, J. A., Golub, M. D., Lund, P. J., Sadtler, P. T., Oby, E. R., Quick, K. M., Ryu, S. I., Tyler-Kabara, E. C., Batista, A. P., Yu, B. M., and Chase, S. M. (2018). Constraints on neural redundancy. *eLife*, 7:1–34.

[75] Huang, C., Ruff, D. A., Pyle, R., Rosenbaum, R., Cohen, M. R., and Doiron, B. (2019). Circuit models of low-dimensional shared variability in cortical networks. *Neuron*, 101:337–348.

[76] Huo, Y., Chen, H., and Guo, Z. V. (2020). Mapping functional connectivity from the dorsal cortex to the thalamus. *Neuron*, 107:1080–1094.

[77] Innes, M. (2018). Don't unroll adjoint: Differentiating ssa-form programs. *CoRR*, abs/1810.07951.

[78] Jacob, P.-Y., Casali, G., Spieser, L., Page, H., Overington, D., and Jeffery, K. (2017). An independent, landmark-dominated head-direction signal in dysgranular retrosplenial cortex. *Nat. Neurosci.*, 20(2):173–175.

[79] Jakob, W. (2012). Numerically stable sampling of the von Mises-Fisher distribution on S^2 (and other tricks).

[80] Jayasumana, S., Hartley, R., Salzmann, M., Li, H., and Harandi, M. (2015). Kernel methods on Riemannian manifolds with Gaussian RBF kernels. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(12):2464–2477.

[81] Jin, X. and Costa, R. M. (2010). Start/stop signals emerge in nigrostriatal circuits during sequence learning. *Nature*, 466:457–462.

[82] Jun, J. J., Steinmetz, N. A., Siegle, J. H., Denman, D. J., Bauza, M., Barbarits, B., Lee, A. K., Anastassiou, C. A., Andrei, A., AydIn, Ç., et al. (2017). Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551(7679):232–236.

[83] Kadmon, J. and Sompolinsky, H. (2015). Transition to chaos in random neuronal networks. *Phys. Rev. X*, 5:041030.

[84] Kandel, E. R., Schwartz, J. H., Jessell, T. M., Siegelbaum, S., Hudspeth, A. J., and Mack, S. (2000). *Principles of neural science*, volume 4. McGraw-hill New York.

[85] Kao, T.-C. and Hennequin, G. (2018). Null ain't dull: new perspectives on motor cortex. *Trends Cog. Sci.*, 22:1069–1071.

[86] Kao, T.-C. and Hennequin, G. (2019). Neuroscience out of control: control-theoretic perspectives on neural circuit dynamics. *Curr. Opin. Neurobiol.*, 58:122–129.

[87] Kaufman, M. T., Churchland, M. M., Ryu, S. I., and Shenoy, K. V. (2014). Cortical activity in the null space: permitting preparation without movement. *Nat. Neurosci.*, 17(3):440–448.

[88] Kaufman, M. T., Seely, J. S., Sussillo, D., Ryu, S. I., Shenoy, K. V., and Churchland, M. M. (2016). The Largest Response Component in the Motor Cortex Reflects Movement Timing but Not Movement Type. *eNeuro*, 3(4):0085–16.2016.

[89] Kawai, R., Markman, T., Poddar, R., Ko, R., Fantana, A. L., Dhawale, A. K., Kampff, A. R., and Ölveczky, B. P. (2015). Motor cortex is required for learning but not for executing a motor skill. *Neuron*, 86(3):800–812.

[90] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[91] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In *ICLR*.

[92] Kobak, D., Brendel, W., Constantinidis, C., Feierstein, C. E., Kepecs, A., Mainen, Z. F., Qi, X.-L., Romo, R., Uchida, N., and Machens, C. K. (2016). Demixed principal component analysis of neural population data. *eLife*, 5:e10989.

[93] Körding, K. P. and Wolpert, D. M. (2004). Bayesian integration in sensorimotor learning. *Nature*, 427(6971):244–247.

[94] Landau, I. D., Egger, R., Dercksen, V. J., Oberlaender, M., and Sompolinsky, H. (2016). The impact of structural heterogeneity on excitation-inhibition balance in cortical networks. *Neuron*, 92:1106–1121.

[95] Lara, A. H., Elsayed, G. F., Zimnik, A. J., Cunningham, J. P., and Churchland, M. M. (2018). Conservation of preparatory neural events in monkey motor cortex regardless of how movement is initiated. *eLife*, 7:e31826.

[96] Lawrence, N. (2005). Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *J. Mach. Learn. Res.*, 6:1783–1816.

[97] Lawrence, N. D. (2004). Gaussian process latent variable models for visualisation of high dimensional data. In *Adv. Neural Inf. Process. Syst.*

[98] Li, N., Daie, K., Svoboda, K., and Druckmann, S. (2016). Robust neuronal dynamics in premotor cortex during motor planning. *Nature*, 532:459–464.

[99] Li, W. and Todorov, E. (2004). Iterative linear quadratic regulator design for nonlinear biological movement systems. *International Conference on Informatics in Control, Automation and Robotics*.

[100] Lillicrap, T. P. and Scott, S. H. (2013). Preference distributions of primary motor cortex neurons reflect control solutions optimized for limb biomechanics. *Neuron*, 77(1):168–179.

[101] Linderman, S., Johnson, M., Miller, A., Adams, R., Blei, D., and Paninski, L. (2017). Bayesian learning and inference in recurrent switching linear dynamical systems. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54, pages 914–922.

[102] Liu, D. C. and Nocedal, J. (1989). On the limited memory bfgs method for large scale optimization. *Math. Prog.*, 45(1-3):503–528.

[103] Logiaco, L., Abbott, L. F., and Escola, S. (2019). A model of flexible motor sequencing through thalamic control of cortical dynamics. *bioRxiv*.

[104] Lou, A., Lim, D., Katsman, I., Huang, L., Jiang, Q., Lim, S.-N., and De Sa, C. (2020). Neural manifold ordinary differential equations. *arXiv preprint arXiv:2006.10254*.

[105] Lusch, B., Kutz, J. N., and Brunton, S. L. (2018). Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.*, 9:1–10.

[106] Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-SNE. *J. Mach. Learn. Res.*, 9:2579–2605.

[107] MacKay, D. J. (1998). Introduction to Gaussian processes. *NATO ASI series. Series F: computer and system sciences*, pages 133–165.

[108] Mallasto, A. and Feragen, A. (2018). Wrapped Gaussian process regression on Riemannian manifolds. In *Proc. IEEE Comput. Soc. Conf. Vis. Pattern Recognit.*, pages 5580–5588.

[109] Mallasto, A., Hauberg, S., and Feragen, A. (2019). Probabilistic Riemannian submanifold learning with wrapped Gaussian process latent variable models. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2368–2377.

[110] Mante, V., Sussillo, D., Shenoy, K. V., and Newsome, W. T. (2013). Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503:78–84.

[111] Mastrogiuseppe, F. and Ostojic, S. (2017). Intrinsically-generated fluctuating activity in excitatory-inhibitory networks. *PLoS Comput. Biol.*, 13:e1005498.

[112] Mastrogiuseppe, F. and Ostojic, S. (2018). Linking connectivity, dynamics, and computations in low-rank recurrent neural networks. *Neuron*, 99:609–623.

[113] Masullo, L., Mariotti, L., Alexandre, N., Freire-Pritchett, P., Boulanger, J., and Tripodi, M. (2019). Genetically defined functional modules for spatial orienting in the mouse superior colliculus. *Curr. Biol.*, 29:2892–2904.

[114] Mathieu, E. and Nickel, M. (2020). Riemannian continuous normalizing flows. *arXiv preprint arXiv:2006.10605*.

[115] Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., and Bethge, M. (2018). Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nat. Neurosci.*, 21(9):1281–1289.

[116] McNamee, D. and Wolpert, D. M. (2019). Internal models in biological control. *Ann. Rev. Contr. Robot. Autonom. Syst.*, 2:339–364.

[117] Merel, J., Aldarondo, D., Marshall, J., Tassa, Y., Wayne, G., and Ölveczky, B. (2019). Deep neuroethology of a virtual rodent. *arXiv preprint arXiv:1911.09451*.

[118] Michaels, J. A., Dann, B., and Scherberger, H. (2016). Neural population dynamics during reaching are better explained by a dynamical system than representational tuning. *PLoS Comput. Biol.*, 12.

[119] Michaels, J. A., Schaffelhofer, S., Agudelo-Toro, A., and Scherberger, H. (2020). A goal-driven modular neural network predicts parietofrontal neural dynamics during grasping. *Proc. Natl. Acad. Sci. U.S.A.*, 117(50):32124–32135.

[120] Moran, D. W. and Schwartz, A. B. (1999). Motor cortical representation of speed and direction during reaching. *J. Neurophsyiol.*, 82(5):2676–2692.

[121] Murphy, B. K. and Miller, K. D. (2009). Balanced amplification: a new mechanism of selective amplification of neural activity patterns. *Neuron*, 61(4):635–648.

[122] Naesseth, C. A., Ruiz, F. J., Linderman, S. W., and Blei, D. M. (2016). Reparameterization gradients through acceptance-rejection sampling algorithms. *arXiv preprint arXiv:1610.05683*.

[123] Nakajima, M. and Halassa, M. M. (2017). Thalamic control of functional cortical connectivity. *Curr. Opin. Neurobiol.*, 44:127–131.

[124] Navarro, A. K., Frellsen, J., and Turner, R. E. (2017). The multivariate generalised von mises distribution: inference and applications. In *AAAI*.

[125] Neal, R. M. (2012). *Bayesian learning for neural networks*, volume 118. Springer.

[126] Nielsen, J. B. (2016). Human spinal motor control. *Annu. Rev. Neurosci.*, 39:81–101.

[127] Omrani, M., Kaufman, M. T., Hatsopoulos, N. G., and Cheney, P. D. (2017). Perspectives on classical controversies about the motor cortex. *J. Neurophysiol.*, 118:1828–1848.

[128] Ozeki, H., Finn, I. M., Schaffer, E. S., Miller, K. D., and Ferster, D. (2009). Inhibitory stabilization of the cortical network underlies visual surround suppression. *Neuron*, 62(4):578–592.

[129] O'Shea, D. J., Kalanithi, P., Ferenczi, E. A., Hsueh, B., Chandrasekaran, C., Goo, W., Diester, I., Ramakrishnan, C., Kaufman, M. T., Ryu, S. I., et al. (2018). Development of an optogenetic toolkit for neural circuit dissection in squirrel monkeys. *Sci. Rep.*, 8:1–20.

[130] Packer, A. M., Russell, L. E., Dalgleish, H. W., and Häusser, M. (2014). Simultaneous all-optical manipulation and recording of neural circuit activity with cellular resolution in vivo. *Nat. Methods*, 12:140–146.

[131] Pandarinath, C., O'Shea, D., Collins, J., Jozefowicz, R., Stavisky, S., Kao, J., Trautmann, E., Kaufman, M., Ryu, S., Hochberg, L., Henderson, J., Shenoy, K., Abbott, L., and Sussillo, D. (2018). Inferring single-trial neural population dynamics using sequential auto-encoders. *Nat. Methods*, 15:805–815.

[132] Paninski, L., Ahmadian, Y., Ferreira, D. G., Koyama, S., Rad, K. R., Vidne, M., Vogelstein, J., and Wu, W. (2010). A new look at state-space models for neural data. *J. Comput. Neurosci.*, 29(1-2):107–126.

[133] Paninski, L. and Cunningham, J. P. (2018). Neural data science: accelerating the experiment-analysis-theory cycle in large-scale neuroscience. *Curr. Opin. Neurobiol.*, 50:232–241.

[134] Paninski, L., Pillow, J., and Lewi, J. (2007). Statistical models for neural encoding, decoding, and optimal stimulus design. *Prog. Brain Res.*, 165:493–507.

[135] Park, J., Coddington, L. T., and Dudman, J. T. (2020). Basal ganglia circuits for action specification. *Annu. Rev. Neurosci.*, 43.

[136] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Adv. Neural Inf. Process. Syst.*, pages 8026–8037.

[137] Paton, J. J. and Buonomano, D. V. (2018). The neural basis of timing: distributed mechanisms for diverse functions. *Neuron*, 98:687–705.

[138] Perich, M. G., Gallego, J. A., and Miller, L. E. (2018). A neural population mechanism for rapid learning. *Neuron*, 100(4):964–976.

[139] Peyrache, A., Lacroix, M. M., Petersen, P. C., and Buzsáki, G. (2015a). Internally organized mechanisms of the head direction sense. *Nat. Neurosci.*, 18(4):569–575.

[140] Peyrache, A., Petersen, P., and Buzsáki, G. (2015b). Extracellular recordings from multi-site silicon probes in the anterior thalamus and subicular formation of freely moving mice. *CRCNS*. Dataset. https://doi.org/10.6080/K0G15XS1.

[141] Press, W. H. (2011). Canonical correlation clarified by singular value decomposition.

[142] Raghu, M., Gilmer, J., Yosinski, J., and Sohl-Dickstein, J. (2017). Svcca: singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *Adv. Neural Inf. Process. Syst.*, pages 6078–6087.

[143] Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian processes for machine learning*. MIT press Cambridge, MA.

[144] Recht, B. (2016). Mechanics of lagrangian. http://www.argmin.net/2016/05/31/mechanics-of-lagrangians/.

[145] Remington, E. D., Narain, D., Hosseini, E. A., and Jazayeri, M. (2018). Flexible sensorimotor computations through rapid reconfiguration of cortical dynamics. *Neuron*, 98(5):1005–1019.

[146] Rey, L. A. P., Menkovski, V., and Portegies, J. W. (2019). Diffusion variational autoencoders. *arXiv preprint arXiv:1901.08991*.

[147] Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. In *31st International Conference on Machine Learning, ICML 2014*, pages 3057–3070.

[148] Rezende, D. J., Papamakarios, G., Racanière, S., Albergo, M. S., Kanwar, G., Shanahan, P. E., and Cranmer, K. (2020). Normalizing flows on tori and spheres. *arXiv preprint arXiv:2022.02428*.

[149] Rikhye, R. V., Gilra, A., and Halassa, M. M. (2018). Thalamic regulation of switching between cortical representations enables cognitive flexibility. *Nat. Neurosci.*, 21:1753–1763.

[150] Rivkind, A. and Barak, O. (2017). Local dynamics in trained recurrent neural networks. *Phys. Rev. Lett.*, 118(25):258101.

[151] Roweis, S. and Ghahramani, Z. (1999). A unifying review of linear gaussian models. *Neural Comput.*, 11:305–345.

[152] Rubin, A., Sheintuch, L., Brande-Eilat, N., Pinchasof, O., Rechavi, Y., Geva, N., and Ziv, Y. (2019). Revealing neural correlates of behavior without behavioral measurements. *Nat. Commun.*, 10:1–14.

[153] Ruder, L. and Arber, S. (2019). Brainstem circuits controlling action diversification. *Annu. Rev. Neurosci.*, 42:485–504.

[154] Russo, A. A., Bittner, S. R., Perkins, S. M., Seely, J. S., London, B. M., Lara, A. H., Miri, A., Marshall, N. J., Kohn, A., Jessell, T. M., et al. (2018). Motor cortex embeds muscle-like commands in an untangled population response. *Neuron*, 97:953–966.

[155] Sadtler, P. T., Quick, K. M., Golub, M. D., Chase, S. M., Ryu, S. I., Tyler-Kabara, E. C., Yu, B. M., and Batista, A. P. (2014). Neural constraints on learning. *Nature*, 512:423–426.

[156] Sanzeni, A., Akitake, B., Goldbach, H. C., Leedy, C. E., Brunel, N., and Histed, M. H. (2019). Inhibition stabilization is a widespread property of cortical networks. *bioRxiv*, page 656710.

[157] Sauerbrei, B. A., Guo, J.-Z., Cohen, J. D., Mischiati, M., Guo, W., Kabra, M., Verma, N., Mensh, B., Branson, K., and Hantman, A. W. (2020). Cortical pattern generation during dexterous movement is input-driven. *Nature*, 577:386–391.

[158] Saxena, S. and Cunningham, J. P. (2019). Towards the neural population doctrine. *Curr. Opin. Neurobiol.*, 55:103–111.

[159] Scott, S. H. (2004). Optimal feedback control and the neural basis of volitional motor control. *Nat. Rev. Neurosci.*, 5(7):534–546.

[160] Scott, S. H. (2012). The computational and neural basis of voluntary motor control and planning. *Trends Cogn. Sci.*, 16:541–549.

[161] Scott, S. H., Cluff, T., Lowrey, C. R., and Takei, T. (2015). Feedback control during voluntary motor actions. *Curr. Opin. Neurobiol.*, 33:85–94.

[162] Seelig, J. D. and Jayaraman, V. (2015). Neural dynamics for landmark orientation and angular path integration. *Nature*, 521(7551):186–191.

[163] Seely, J. S., Kaufman, M. T., Ryu, S. I., Shenoy, K. V., Cunningham, J. P., and Churchland, M. M. (2016). Tensor analysis reveals distinct population structure that parallels the different computational roles of areas m1 and v1. *PLoS Comput. Biol.*, 12.

[164] Semedo, J. D., Zandvakili, A., Machens, C. K., Yu, B. M., and Kohn, A. (2019). Cortical areas interact through a communication subspace. *Neuron*, 102:1–11.

[165] Sheahan, H. R., Franklin, D. W., and Wolpert, D. M. (2016). Motor planning, not execution, separates motor memories. *Neuron*, 92:773–779.

[166] Shenoy, K. V., Sahani, M., and Churchland, M. M. (2013). Cortical control of arm movements: a dynamical systems perspective. *Ann. Rev. Neurosci.*, 36:337–359.

[167] Shepard, R. N. and Metzler, J. (1971). Mental rotation of three-dimensional objects. *Science*, 171:701–703.

[168] Skogestad, S. and Postlethwaite, I. (2007). *Multivariable feedback control: analysis and design*, volume 2. Wiley New York.

[169] Sohn, H., Narain, D., Meirhaeghe, N., and Jazayeri, M. (2019). Bayesian computation through cortical latent dynamics. *Neuron*, 103:934–947.

[170] Sola, J., Deray, J., and Atchuthan, D. (2018). A micro Lie theory for state estimation in robotics. *arXiv preprint arXiv:1812.01537*.

[171] Straub, J. (2017). Bayesian inference with the von-Mises-Fisher distribution in 3d.

[172] Stringer, C., Pachitariu, M., Steinmetz, N., Reddy, C. B., Carandini, M., and Harris, K. D. (2019). Spontaneous behaviors drive multidimensional, brainwide activity. *Science*, 364(6437).

[173] Stroud, J. P., Porter, M. A., Hennequin, G., and Vogels, T. P. (2018). Motor primitives in space and time via targeted gain modulation in cortical networks. *Nat. Neurosci.*, 21:1774–1783.

[174] Sussillo, D. (2014). Neural circuits as computational dynamical systems. *Curr. Op. Neurobiol.*, 25:156–163.

[175] Sussillo, D. and Abbott, L. F. (2009). Generating coherent patterns of activity from chaotic neural networks. *Neuron*, 63:544–557.

[176] Sussillo, D. and Barak, O. (2013a). Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Comput.*, 25:626–649.

[177] Sussillo, D. and Barak, O. (2013b). Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. *Neural Comput.*, 25:626–649.

[178] Sussillo, D., Churchland, M. M., Kaufman, M. T., and Shenoy, K. V. (2015). A neural network that finds a naturalistic solution for the production of muscle activity. *Nat. Neurosci.*, 18:1025–1033.

[179] Svoboda, K. and Li, N. (2018). Neural mechanisms of movement planning: motor cortex and beyond. *Curr. Op. Neurobiol.*, 49:33–41.

[180] Tanji, J. and Evarts, E. V. (1976). Anticipatory activity of motor cortex neurons in relation to direction of an intended movement. *J. Neurophysiol.*, 39(5):1062–1068.

[181] Titsias, M. K. (2009). Variational learning of inducing variables in sparse Gaussian processes. In *J. Mach. Learn. Res.*, volume 5, pages 567–574.

[182] Titsias, M. K. and Lawrence, N. D. (2010). Bayesian Gaussian process latent variable model. In *J. Mach. Learn. Res.*, volume 9, pages 844–851.

[183] Todorov, E. (2000a). Direct cortical control of muscle activation in voluntary arm movements: a model. *Nat. Neurosci.*, 3(4):391–398.

[184] Todorov, E. (2000b). Direct cortical control of muscle activation in voluntary arm movements: a model. *Nat. Neurosci.*, 3:391–398.

[185] Todorov, E. (2004). Optimality principles in sensorimotor control. *Nat. Neurosci.*, 7(9):907.

[186] Todorov, E., Erez, T., and Tassa, Y. (2012). MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE.

[187] Todorov, E. and Jordan, M. I. (2002). Optimal feedback control as a theory of motor coordination. *Nat. Neurosci.*, 5(11):1226–1235.

[188] Tosi, A., Hauberg, S., Vellido, A., and Lawrence, N. D. (2014). Metrics for probabilistic geometries. *arXiv preprint arXiv:1411.7432*.

[189] Trefethen, L. N. and Embree, M. (2005). *Spectra and pseudospectra: the behavior of nonnormal matrices and operators*. Princeton University Press.

[190] Tsodyks, M. V., Skaggs, W. E., Sejnowski, T. J., and McNaughton, B. L. (1997). Paradoxical effects of external modulation of inhibitory interneurons. *J. Neurosci.*, 17(11):4382–4388.

[191] Turner-Evans, D. B., Jensen, K. T., Ali, S., Paterson, T., Sheridan, A., Ray, R. P., Wolff, T., Lauritzen, J. S., Rubin, G. M., Bock, D. D., and Jayaraman, V. (2020). The neuroanatomical ultrastructure and function of a biological ring attractor. *Neuron*, 108:145–163.

[192] Ulrich, G. (1984). Computer generation of distributions on the M-sphere. *J. Roy. Stat. Soc. Appl. Stats.*, 33(2):158–163.

[193] Urtasun, R., Fleet, D. J., Geiger, A., Popović, J., Darrell, T. J., and Lawrence, N. D. (2008). Topologically-constrained latent variable models. In *ICML*, pages 1080–1087.

[194] Vogels, T. P. and Abbott, L. (2009). Gating multiple signals through detailed balance of excitation and inhibition in spiking networks. *Nat. Neurosci.*, 12:483–491.

[195] Vogels, T. P., Sprekeler, H., Zenke, F., Clopath, C., and Gerstner, W. (2011). Inhibitory plasticity balances excitation and inhibition in sensory pathways and memory networks. *Science*, 334:1569–1573.

[196] Vyas, S., Golub, M. D., Sussillo, D., and Shenoy, K. V. (2020). Computation through neural population dynamics. *Annu. Rev. Neurosci.*, 43:249–275.

[197] Wang, J., Narain, D., Hosseini, E. A., and Jazayeri, M. (2018). Flexible timing by temporal scaling of cortical responses. *Nat. Neurosci.*, 21:102–110.

[198] Wang, L. (2017). Owl: A general-purpose numerical library in OCaml. *arXiv preprint arXiv:1707.09616*.

[199] Wang, P. Z. and Wang, W. Y. (2019). Riemannian normalizing flow on variational Wasserstein autoencoder for text modeling. *arXiv preprint arXiv:1904.02399*.

[200] Wang, X.-J. (2012). Neural dynamics and circuit mechanisms of decision-making. *Curr. Op. Neurobiol.*, 22(6):1039–1046.

[201] Wärnberg, E. and Kumar, A. (2019). Perturbing low dimensional activity manifolds in spiking neuronal networks. *PLoS Comput. Biol.*, 15:e1007074.

[202] Weisenburger, S. and Vaziri, A. (2018). A guide to emerging technologies for large-scale and whole-brain optical imaging of neuronal activity. *Annu. Rev. Neurosci.*, 41:431–452.

[203] White, O. L., Lee, D. D., and Sompolinsky, H. (2004). Short-term memory in orthogonal neural networks. *Phys. Rev. Lett.*, 92:1–4.

[204] Wilson, J. J., Alexandre, N., Trentin, C., and Tripodi, M. (2018). Three-dimensional representation of motor space in the mouse superior colliculus. *Curr. Biol.*, 28(11):1744–1755.e12.

[205] Wolpert, D. M. (1997). Computational approaches to motor control. *Trends Cog. Sci.*, 1(6):209–216.

[206] Wolpert, D. M. and Ghahramani, Z. (2000). Computational principles of movement neuroscience. *Nat. Neurosci.*, 3:1212–1217.

[207] Wolpert, D. M., Ghahramani, Z., and Jordan, M. I. (1995). An internal model for sensorimotor integration. *Science*, 269:1880–1882.

[208] Wolpert, D. M., Miall, R. C., and Kawato, M. (1998). Internal models in the cerebellum. *Trends Cog. Sci.*, 2(9):338–347.

[209] Wu, A., Pashkovski, S., Datta, S. R., and Pillow, J. W. (2018). Learning a latent manifold of odor representations from neural responses in piriform cortex. In *Adv. Neural Inf. Process. Syst.*, pages 5378–5388.

[210] Wu, A., Roy, N. A., Keeley, S., and Pillow, J. W. (2017). Gaussian process based nonlinear latent structure discovery in multivariate spike train data. In *Adv. Neural Inf. Process. Syst.*, pages 3497–3506.

[211] Yamins, D. L. and DiCarlo, J. J. (2016). Using goal-driven deep learning models to understand sensory cortex. *Nat. Neurosci.*, 19(3):356–365.

[212] Yan, G., Vértes, P. E., Towlson, E. K., Chew, Y. L., Walker, D. S., Schafer, W. R., and Barabási, A.-L. (2017). Network control principles predict neuron function in the Caenorhabditis elegans connectome. *Nature*, 550(7677):519–523.

[213] Yu, B. M., Cunningham, J. P., Santhanam, G., Ryu, S. I., Shenoy, K. V., and Sahani, M. (2009). Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *J. Neurophysiol.*, 102(1):614–635.

[214] Zenke, F., Gerstner, W., and Ganguli, S. (2017). The temporal paradox of Hebbian learning and homeostatic plasticity. *Curr. Op. Neurobiol.*, 43:166–176.

[215] Zhang, H., Li, Y., and Hu, X. (2019). Inverse optimal control for finite-horizon discrete-time linear quadratic regulator under noisy output. In *IEEE 58th Conf. Decis. Contr.*, pages 6663–6668. IEEE.

[216] Zimnik, A. J. and Churchland, M. M. (2021). Independent generation of sequence elements by motor cortex. *Nat. Neurosci.*, pages 412–424.